Automata and Logic

Radu Iosif
Verimag/CNRS (Grenoble, France)
Preliminaries
**Words**

An *alphabet* is a **finite** non-empty set of symbols $\Sigma = \{a, b, c, \ldots\}$.

A *finite word* of length $n$ over $\Sigma$ is a sequence $w = a_1a_2 \ldots a_n$, where $a_i \in \Sigma$, for all $1 \leq i \leq n$. The length of the word $w$ is denoted by $|w|$. The *empty word* is denoted by $\epsilon$, i.e. $|\epsilon| = 0$.

An *infinite word* is an infinite sequence of elements of $\Sigma$, i.e. a function $w : \mathbb{N} \rightarrow \Sigma$.

$\Sigma^*$ ($\Sigma^\omega$) is the set of all finite (infinite) words over $\Sigma$.

The *concatenation* of two words $w$ and $u$ is denoted as $wu$. A *prefix* $u$ of $w$ ($u \leq w$) is any word $u \in \Sigma^*$ s.t. there exists $v \in \Sigma^*$ such that $uv = w$. 
Trees

A **prefix-closed** set $S \in \Sigma^*$ is a set such that for all $w \in S$ and $u \in \Sigma^*$, $u \leq w \Rightarrow u \in S$.

A **tree** over $\Sigma$ is a partial function $t : \mathbb{N}^* \rightarrow \Sigma$ such that $\text{dom}(t)$ is a prefix-closed set.

A tree $t$ is said to be **finite-branching** iff for all $p \in \text{dom}(t)$, the number of children of $p$ is finite. A tree $t$ is said to be **finite** if $\text{dom}(t)$ is finite.

**Lemma 1 (König)** A finitely branching tree is infinite if and only if it has an infinite path.
Ranked Trees

A ranked alphabet \( \langle \Sigma, \# \rangle \) is a set of symbols together with a function \( \# : \Sigma \to \mathbb{N} \). For \( f \in \Sigma \), the value \( \#(f) \) is said to be the arity of \( f \).

A ranked tree \( t \) over \( \Sigma \) is a partial function \( t : \mathbb{N}^* \to \Sigma \) that satisfies the following conditions:

- \( \text{dom}(t) \) is a finite prefix-closed subset of \( \mathbb{N}^* \), and
- for each \( p \in \text{dom}(t) \):
  
  \[
  \text{if } \#(t(p)) = n > 0 \text{ then } \{i \mid pi \in \text{dom}(t)\} = \{1, \ldots, n\}
  \]

A finite tree over a ranked alphabet is also called a term.
First Order Logic
Syntax

The alphabet of FOL consists of the following symbols:

- **predicate symbols:** $p_1, p_2, \ldots, =$
- **function symbols:** $f_1, f_2, \ldots$
- **constant symbols:** $c_1, c_2, \ldots$
- **first-order variables:** $x, y, z, \ldots$
- **connectives:** $\lor, \land, \to, \leftrightarrow, \neg, \bot, \forall, \exists$

**Note:** The alphabet of a logic may be, in principle, infinite (yet countable). Here we will consider only logics defined over finite alphabets.
**Syntax**

The set of *first-order terms* is defined inductively:

- any constant symbol $c$ is a term,
- any first-order variable $x$ is a term,
- if $t_1, t_2, \ldots, t_n$ are terms and $f$ is a function symbol of arity $n > 0$, then $f(t_1, t_2, \ldots, t_n)$ is a term,
- nothing else is a term.

A term with no variable is said to be a *ground term*. An *atomic proposition* is any proposition of the form $p(t_1, \ldots, p_n)$ or $t_1 = t_2$, where $t_1, t_2, \ldots, t_n$ are terms.
Syntax

The set of first-order formulae is defined inductively:

- \( \bot \) and \( \top \) are formulae,
- \( p \) is a formula, if \( \#(p) = 0 \),
- if \( t_1, t_2, \ldots, t_n \) are terms and \( p \) is a predicate symbol of arity \( n > 0 \), then \( p(t_1, t_2, \ldots, t_n) \) is a formula,
- if \( t_1, t_2 \) are terms, then \( t_1 = t_2 \) is a formula,
- if \( \varphi \) and \( \psi \) are formulae, then \( \varphi \bullet \psi, \neg \varphi, \forall x . \varphi \) and \( \exists x . \varphi \) are formulae, for \( \bullet \in \{ \lor, \land, \rightarrow, \leftrightarrow \} \),
- nothing else is a formula.

The *language* of logic FOL is the set of formulae, denoted as \( \mathcal{L}(FOL) \).
\[ x = y \]
\[ \forall x \forall y . \ x = y \iff y = x \]
\[ \exists x (\forall y . p(x, y)) \rightarrow q(x) \]
\[ \forall x . p(x) \rightarrow q(f(x)) \]
\[ \forall x \exists y . f(x) = y \land (\forall z . f(z) = y \rightarrow z = x) \]
**FOL Formulae**

The *size* of a formula is the number of subformulae it contains, in other words, the number of nodes in the syntax tree representing the formula. The size of \( \varphi \) is denoted as \(|\varphi|\).

The variables within the scope of a quantifier are said to be *bound*. The variables that are not bound are said to be *free*. We denote by \( FV(\varphi) \) the set of free variables in \( \varphi \). If \( FV(\varphi) = \emptyset \) then \( \varphi \) is said to be a *sentence*.

**Example 1** \( FV(\forall x \cdot x = y \land x = z \rightarrow p(x)) = \{y, z\} \)

If \( x \in FV(\varphi) \), we denote by \( \varphi[t/x] \) the formula obtained from \( \varphi \) by substituting \( x \) with the term \( t \).
**Semantics**

A *structure* is a tuple \( m = \langle U, \bar{p}_1, \bar{p}_2, \ldots, \bar{f}_1, \bar{f}_2, \ldots \rangle \), where:

- \( U \) is a (possible infinite) set called the *universe*,
- \( \bar{p}_i \subseteq U^{\#(p_i)}, i = 1, 2, \ldots \) are the *predicates*,
- \( \bar{f}_i : U^{\#(f_i)} \to U, i = 1, 2, \ldots \) are the *functions*.

The elements of the universe are called *individuals*, denoted by \( \bar{c}_1, \bar{c}_2, \ldots \).

**Note:** Each individual \( \bar{c} \) has a corresponding constant symbol \( c \). However, not all constant symbols from the set \( \{ c \mid \bar{c} \in U \} \) need to be in the alphabet of the logic.

**Example:** The set of natural numbers \( \langle \mathbb{N}, 0, +, \cdot, S \rangle \) where \( S(x) = x + 1 \). □
**Semantics**

The *interpretation* of a ground term $t$ in a structure $m$ is denoted as $t^m \in U$:

$$
\begin{align*}
    c^m &= \bar{c} \in U \\
    f(t_1, \ldots, t_n)^m &= \bar{f}(t_1^m, \ldots, t_n^m)
\end{align*}
$$

The *meaning* of a sentence $\varphi$ in a structure $m$ is denoted as $[\varphi]_m \in \{\text{true, false}\}$:

- $[\bot]_m = \text{false}$
- $[p(t_1, \ldots, t_n)]_m = \text{true}$ iff $\langle t_1^m, \ldots, t_n^m \rangle \in \bar{p}$
- $[t_1 = t_2]_m = \text{true}$ iff $t_1^m = t_2^m$
- $[\neg \varphi]_m = \text{true}$ iff $[\varphi]_m = \text{false}$
- $[\varphi \land \psi]_m = \text{true}$ iff $[\varphi]_m = [\psi]_m = \text{true}$
- $[\exists x \cdot \varphi]_m = \text{true}$ iff $[\varphi[c/x]]_m = \text{true}$, for some individual $\bar{c} \in U$
Semantics

Derived meanings:

\[
\begin{align*}
[\varphi \lor \psi]_m &= [\neg(\varphi \land \psi)]_m \\
[\varphi \rightarrow \psi]_m &= [\neg\varphi \lor \psi]_m \\
[\varphi \leftrightarrow \psi]_m &= [(\varphi \rightarrow \psi) \land (\psi \rightarrow \varphi)]_m \\
[\forall x . \varphi] &= [\neg\exists x . \neg\varphi]_m
\end{align*}
\]

If \([\varphi]_m = \text{true}\) we say that \(m\) is a model of \(\varphi\), denoted as \(m \models \varphi\). If \(m \models \varphi\) for all structures \(m\), we say that \(\varphi\) is valid, denoted as \(\models \varphi\). If \(\varphi\) has at least one model, we say that it is satisfiable.

**SAT**: Given \(\varphi\) is it satisfiable? (Hilbert’s Entscheidungsproblem)
Examples

Let $\leq$ be a binary predicate symbol, and $m = \langle U, \leq \rangle$ be a structure. $m$ is a partially ordered set if $m \models \varphi_1 \land \varphi_2$, where:

$$\varphi_1 : \forall x \forall y . x \leq y \land y \leq x \leftrightarrow x = y$$

$$\varphi_2 : \forall x \forall y \forall z . x \leq y \land y \leq z \to x \leq z$$

Notice that $\models \varphi_1 \to \forall x . x \leq x$. $m$ is a linearly ordered set if $m \models \varphi_1 \land \varphi_2 \land \varphi_3$, where:

$$\varphi_3 : \forall x \forall y . x \leq y \lor y \leq x$$
Normal Forms

A formula $\varphi \in \mathcal{L}(FOL)$ is said to be quantifier-free iff it contains no quantifiers.

A quantifier-free formula $\varphi \in \mathcal{L}(FOL)$ is said to be in negation normal form (NNF) iff the only subformulae appearing under negation are atomic propositions.

A formula $\varphi \in \mathcal{L}(FOL)$ is said to be in prenex normal form (PNF) iff

$$\varphi = Q_1x_1Q_2x_2 \cdots Q_nx_n \cdot \psi(x_1, x_2, \ldots, x_n)$$

where $Q_i \in \{\exists, \forall\}$ and $\psi$ is a quantifier-free formula. Sometimes $\psi$ is said to be the matrix of $\varphi$. 
**Normal Forms**

A quantifier-free formula $\varphi \in \mathcal{L}(FOL)$ is said to be in *disjunctive normal form* (DNF) iff $\varphi = \bigvee_i \bigwedge_j \lambda_{ij}$, where $\lambda_{ij}$ are either atomic propositions or negations of atomic propositions.

A quantifier-free formula $\varphi \in \mathcal{L}(FOL)$ is said to be in *conjunctive normal form* (CNF) iff $\varphi = \bigwedge_i \bigvee_j \lambda_{ij}$, where $\lambda_{ij}$ are either atomic propositions or negations of atomic propositions.
FOL on Finite Words

Let $\Sigma = \{a, b, \ldots\}$ be an alphabet and $w = a_0a_1 \ldots a_{n-1}$ be a finite word.

The structure corresponding to $w$ is $m_w = \langle \text{dom}(w), \{\bar{p}_a\}_{a \in \Sigma}, \leq \rangle$, where:

- $\text{dom}(w) = \{0, 1, \ldots, n - 1\}$,
- $\bar{p}_a = \{x \in \text{dom}(w) \mid w(x) = a\}$,
- $x \leq y$ iff $x \leq y$.

$m_{abbaab} = \langle \{0, \ldots, 5\}, \bar{p}_a = \{0, 3, 4\}, \bar{p}_b = \{1, 2, 5\}, \leq \rangle$

**Ex:** Define the successor relation $S(x, y) : 0 \leq x, y \leq n - 1 \land x + 1 = y$. 
FOL on Infinite Words

Let \( w : \mathbb{N} \rightarrow \Sigma \) be an infinite word.

The structure corresponding to \( w \) is \( m_w = \langle \mathbb{N}, \{\bar{p}_a\}_{a \in \Sigma}, \leq \rangle \).

\[
m_{(ab)\omega} = \langle \mathbb{N}, \bar{p}_a = \{2k \mid k \in \mathbb{N}\}, \bar{p}_b = \{2k + 1 \mid k \in \mathbb{N}\}, \leq \rangle
\]
FOL on Finite Trees

Let $\Sigma = \{f, g, \ldots\}$ be an alphabet and $t : \mathbb{N}^* \to \Sigma$ be a finite tree over $\Sigma$.

The structure corresponding to $t$ is $m_t = \langle \text{dom}(t), \{\bar{p}_f\}_{f \in \Sigma}, \preceq, \{s_n\}_{n \in \mathbb{N}} \rangle$
where:

- $\bar{p}_f = \{p \in \text{dom}(t) \mid t(p) = f\}$,
- $\preceq$ is the prefix order on $\mathbb{N}^*$,
- $s_n(p) = pn$ for any $n \in \mathbb{N}$, is the $n$-th successor function

Note: If $p$ doesn't have an $n$-th child, we assume $s_n(p) = p$.

$m_{f(f(g,g),g)} = \langle \{\epsilon, 0, 1, 00, 01\}, \bar{p}_f = \{\epsilon, 0\}, \bar{p}_g = \{00, 01, 1\}, \preceq, \{s_n\}_{n \in \mathbb{N}} \rangle$. 
FOL on Infinite Trees

Let \( t : \mathbb{N}^* \to \Sigma \) be an infinite tree over \( \Sigma \).

The structure corresponding to \( t \) is \( m_t = \langle \mathbb{N}^*, \{\bar{p}_f\}_{f \in \Sigma}, \preceq, \{s_n\}_{n \in \mathbb{N}} \rangle \).

The lexicographic order on \( \mathbb{N}^* \) is defined as follows:

\[
x \preceq y : x \leq y \lor \exists z . \ s_0(z) \leq x \land s_1(z) \leq y
\]
Monadic Second Order Logic
Syntax

The alphabet of MSOL consists of:

- all first-order symbols
- set variables: \( X, Y, Z, \ldots \)

The set of MSOL terms consists of all first-order terms and set variables. The set of MSOL formulae consists of:

- all first-order formulae, i.e. \( \mathcal{L}(FOL) \subseteq \mathcal{L}(MSOL) \),
- if \( t \) is a term and \( X \) is a set variable, then \( X(t) \) is a formula,
- if \( \varphi \) and \( \psi \) are formulae, then \( \varphi \cdot \psi, \neg \varphi, \forall x . \varphi, \exists x . \varphi, \forall X . \varphi \) and \( \exists X . \varphi \) are formulae, for \( \cdot \in \{ \lor, \land, \rightarrow, \leftrightarrow \} \).

\( C(t) \) and \( X(t) \) are sometimes written \( t \in C \) and \( t \in X \).
Examples

\[ \exists X \forall x . X(x) \]
\[ \forall x . X(x) \rightarrow Y(x) \]
\[ \forall Y . ((\forall x . Y(x) \rightarrow X(x)) \land \exists x . X(x) \land \neg Y(x)) \rightarrow \forall x . \neg Y(x) \]
Semantics

Let $m = \langle U, \bar{p}_1, \bar{p}_2, \ldots, \bar{f}_1, \bar{f}_2, \ldots \rangle$ be a structure. The meaning of a sentence $\varphi$ in a structure $m$ is defined by all FOL rules, and in addition by:

$$[[\exists X . \varphi]]_m = \text{true} \iff [[\varphi[p/X]]]_m = \text{true}, \text{ for some set } \bar{p} \subseteq U, \#(p) = 1$$

Example: Define all partitions $\langle X, Y \rangle$ of $Z$:

$$\text{partition}(X, Y, Z) : (\forall x \forall y . X(x) \wedge Y(y) \rightarrow \neg x = y) \wedge (\forall x . Z(x) \leftrightarrow X(x) \vee Y(x))$$

□
Let $\Sigma = \{a, b, \ldots\}$ be a finite alphabet. The alphabet of the sequential calculus is composed of:

- the infix predicate symbol $\leq$ denoting the linear ordering of positions,
- the set constants $\{p_a \mid a \in \Sigma\}$; $p_a$ denotes the set of positions of $a$.
- the first and second order variables and connectives.

(Weak) indicates that quantification is over finite sets only.
Examples

- The formula $\text{len}(x) : \forall y . y \leq x$ defines the length of a finite word and is unsatisfiable on infinite words.

- The set of positions of a word is defined by the formula $\text{pos}(X) : \forall x . X(x)$.

- The set of even positions is defined by $\text{even}(X) : \exists Y, Z . \text{pos}(Z) \land \text{partition}(X, Y, Z) \land \forall x, y . X(x) \land S(x, y) \rightarrow Y(y) \land \forall x, y . Y(x) \land S(x, y) \rightarrow Y(x)$.

- The set of all words having $a$’s on even positions is the set of models of the sentence: $\exists X . \text{even}(X) \land \forall x . X(x) \rightarrow p_a(x)$. 
MSOL on Trees: (W)SωS

Let $\Sigma = \{a, b, \ldots \}$ be a tree alphabet. The alphabet of (W)SωS is:

- the function symbols $\{s_i \mid i \in \mathbb{N}\}$; $s_i(x)$ denotes the $i$-th successor of $x$
- the set constants $\{p_a \mid a \in \Sigma\}$; $p_a$ denotes the set of positions of $a$
- the first and second order variables and connectives.
Examples

Let us consider binary trees, i.e. the alphabet of S2S.

- The formula $\text{closed}(X) : \forall x . X(x) \rightarrow X(S_0(x)) \land X(S_1(x))$ denotes the fact that $X$ is a downward-closed set.

- The prefix ordering on tree positions is defined by $x \leq y : \forall X . \text{closed}(X) \land X(x) \rightarrow X(y)$.

- The root of a tree is defined by $\text{root}(x) : \forall y . x \leq y$. 
Automata on Finite Words
**Definition**

A non-deterministic finite automaton (NFA) over $\Sigma$ is a tuple $A = \langle S, I, T, F \rangle$ where:

- $S$ is a finite set of *states*,
- $I \subseteq S$ is a set of *initial states*,
- $T \subseteq S \times \Sigma \times S$ is a *transition relation*,
- $F \subseteq S$ is a set of *final states*.

We denote $T(s, \alpha) = \{ s' \in S \mid (s, \alpha, s') \in T \}$. When $T$ is clear from the context we denote $(s, \alpha, s') \in T$ by $s \xrightarrow{\alpha} s'$. 
Determinism and Completeness

**Definition 1** An automaton $A = \langle S, I, T, F \rangle$ is deterministic (DFA) iff $\|I\| = 1$ and, for each $s \in S$ and for each $\alpha \in \Sigma$, $\|T(s, \alpha)\| \leq 1$.

If $A$ is deterministic we write $T(s, \alpha) = s'$ instead of $T(s, \alpha) = \{s'\}$.

**Definition 2** An automaton $A = \langle S, I, T, F \rangle$ is complete iff for each $s \in S$ and for each $\alpha \in \Sigma$, $\|T(s, \alpha)\| \geq 1$. 
Runs and Acceptance Conditions

Given a finite word $w \in \Sigma^*$, $w = \alpha_1\alpha_2 \ldots \alpha_n$, a run of $A$ over $w$ is a finite sequence of states $s_1, s_2, \ldots, s_n, s_{n+1}$ such that $s_1 \in I$ and $s_i \xrightarrow{\alpha_i} s_{i+1}$ for all $1 \leq i \leq n$.

The existence of a run between $s_1$ and $s_{n+1}$ is denoted as $s_1 \xrightarrow{w} s_{n+1}$.

The run is said to be accepting iff $s_{n+1} \in F$. If $A$ has an accepting run over $w$, then we say that $A$ accepts $w$.

The language of $A$, denoted $\mathcal{L}(A)$ is the set of all words accepted by $A$.

A set of words $S \subseteq \Sigma^*$ is rational if there exists an automaton $A$ such that $S = \mathcal{L}(A)$. 

Determinism, Completeness, again

Proposition 1 If $A$ is deterministic, then it has at most one run for each input word.

Proposition 2 If $A$ is complete, then it has at least one run for each input word.
Determinization

Theorem 1  For every NFA $A$ there exists a DFA $A_d$ such that
$L(A) = L(A_d)$.

Let $A_d = \langle 2^S, \{I\}, T_d, \{G \subseteq S \mid G \cap F \neq \emptyset\} \rangle$, where

$$(S_1, \alpha, S_2) \in T_d \iff S_2 = \{s' \mid \exists s \in S_1 . (s, \alpha, s') \in T\}$$
Lemma 2  For every NFA $A$ there exists a complete NFA $A_c$ such that $\mathcal{L}(A) = \mathcal{L}(A_c)$.

Let $A_c = \langle S \cup \{\sigma\}, I, T_c, F \rangle$, where $\sigma \notin S$ is a new sink state. The transition relation $T_c$ is defined as:

$$\forall s \in S \forall \alpha \in \Sigma . \ (s, \alpha, \sigma) \in T_c \iff \forall s' \in S . \ (s, \alpha, s') \notin T$$

and $\forall \alpha \in \Sigma . \ (\sigma, \alpha, \sigma) \in T_c$. 
**Closure Properties**

**Theorem 2** Let $A_1 = \langle S_1, I_1, T_1, F_1 \rangle$ and $A_2 = \langle S_2, I_2, T_2, F_2 \rangle$ be two NFA. There exists automata $\bar{A}_1$, $A_\cup$ and $A_\cap$ that recognize the languages $\Sigma^* \setminus \mathcal{L}(A_1)$, $\mathcal{L}(A_1) \cup \mathcal{L}(A_2)$, and $\mathcal{L}(A_1) \cap \mathcal{L}(A_2)$ respectively.

Let $A' = \langle S', I', T', F' \rangle$ be the complete deterministic automaton such that $\mathcal{L}(A_1) = \mathcal{L}(A')$, and $\bar{A}_1 = \langle S', I', T', S' \setminus F' \rangle$.

Let $A_\cup = \langle S_1 \cup S_2, I_1 \cup I_2, T_1 \cup T_2, F_1 \cup F_2 \rangle$.

Let $A_\cap = \langle S_1 \times S_2, I_1 \times I_2, T_\cap, F_1 \times F_2 \rangle$ where:

$$(\langle s_1, t_1 \rangle, \alpha, \langle s_2, t_2 \rangle) \in T_\cap \iff (s_1, \alpha, s_2) \in T_1 \text{ and } (t_1, \alpha, t_2) \in T_2$$
On the Exponential Blowup of Complementation

Theorem 3 For every $n \in \mathbb{N}$, $n \geq 1$, there exists an automaton $A$, with size$(A) = n + 1$ such that no deterministic automaton with less than $2^n$ states recognizes the complement of $L(A)$.

Let $\Sigma = \{a, b\}$ and $L = \{uav \mid u, v \in \Sigma^*, |v| = n - 1\}$.

There exists a NFA with exactly $n + 1$ states which recognizes $L$.

Suppose that $B = \langle S, \{s_0\}, T, F \rangle$, is a DFA with $|S| < 2^n$ that accepts $\Sigma^* \setminus L$.

Let $X = \{w \in \Sigma^* \mid |w| = n\}$. Since $|X| = 2^n$ and $|S| < 2^n$ then

$\exists uav_1, ubv_2 \in X$, $s \in S$ . $s_0 \xrightarrow{uav_1} s$ and $s_0 \xrightarrow{ubv_2} s$
Since $B$ is deterministic, there is at most one $s' \in S$ such that $s \xrightarrow{u} s'$.

Since $|uav_1| = n$, then $uav_1u \in L \Rightarrow uav_1u \not\in \mathcal{L}(B)$, then $s' \not\in F$.

On the other hand, $ubv_2u \not\in L \Rightarrow ubv_2u \in \mathcal{L}(B)$, then $s' \in F$.

Contradiction. □
Projections

Let the input alphabet $\Sigma = \Sigma_1 \times \Sigma_2$. Any word $w \in \Sigma^*$ can be uniquely identified to a pair $\langle w_1, w_2 \rangle \in \Sigma_1^* \times \Sigma_2^*$ such that $|w_1| = |w_2| = |w|$.

The projection operations are

$pr_1(L) = \{u \in \Sigma_1^* \mid \langle u, v \rangle \in L, \text{ for some } v \in \Sigma_2^*\}$ and

$pr_2(L) = \{v \in \Sigma_2^* \mid \langle u, v \rangle \in L, \text{ for some } u \in \Sigma_1^*\}$.

**Theorem 4** If the language $L \subseteq (\Sigma_1 \times \Sigma_2)^*$ is rational, then so are the projections $pr_i(L)$, for $i = 1, 2$. 
Remark

The operations of union, intersection and complement correspond to the boolean $\lor$, $\land$ and $\neg$.

The projection corresponds to the first-order existential quantifier $\exists x$. 
**Congruences**

**Definition 3** An equivalence relation $\simeq \subseteq \Sigma^* \times \Sigma^*$ is said to be a left-congruence iff for all $u, v, w \in \Sigma^*$ we have $u \simeq v \Rightarrow wu \simeq vw$.

**Definition 4** An equivalence relation $\simeq \subseteq \Sigma^* \times \Sigma^*$ is said to be a right-congruence iff for all $u, v, w \in \Sigma^*$ we have $u \simeq v \Rightarrow uw \simeq vw$.

**Definition 5** An equivalence relation $\simeq \subseteq \Sigma^* \times \Sigma^*$ is said to be a congruence iff it is both a left- and a right-congruence.
An Automata Congruence

Let $A = \langle S, I, T, F \rangle$ be an automaton over the alphabet $\Sigma^*$.

Define the relation $\sim_A \subseteq \Sigma^* \times \Sigma^*$ as:

$$u \sim_A v \iff [\forall s, s' \in S . s \xrightarrow{u} s' \iff s \xrightarrow{v} s']$$

$\sim_A$ is an equivalence relation of finite index (why?)

**Lemma 3** $\sim_A$ is a congruence.
Some Language Congruences

Let $L \subseteq \Sigma^*$ be a language (non-necessarily rational).

Define the relations:

1. $u \sim_L^l v \iff [\forall w \in \Sigma^* . wu \in L \iff vw \in L]$

2. $u \sim_L^r v \iff [\forall w \in \Sigma^* . uw \in L \iff vw \in L]$

3. $u \sim_L v \iff [\forall w, w' \in \Sigma^* . wuw' \in L \iff wvw' \in L]$

Lemma 4

- $\sim_L^l$ is a left congruence
- $\sim_L^r$ is a right congruence
- $\sim_L$ is a congruence
The Myhill-Nerode Theorem

Theorem 5  A language $L \subseteq \Sigma^*$ is rational iff $\sim_L$ is of finite index.

$\Rightarrow$ Suppose $L = \mathcal{L}(A)$ for some automaton $A$.

$\sim_A$ is of finite index

for all $u, v \in \Sigma^*$ we have $u \sim_A v \Rightarrow u \sim_L v$

The index of $\sim_L$ is less than the index of $\sim_A$, thus finite.
The Myhill-Nerode Theorem

“⇐” If $\sim_L$ is an equivalence relation of finite index, then so is $\sim^r_L$, and let $[u]_r$ denote the equivalence class of $u \in \Sigma^*$ w.r.t. $\sim^r_L$.

\[ A = \langle S, I, T, F \rangle, \text{ where:} \]

- \( S = \{ [u]_r \mid u \in \Sigma^* \} \),
- \( I = [\epsilon]_r \),
- \( [u]_r \xrightarrow{\alpha} [v]_r \iff u\alpha \sim^r_L v \),
- \( F = \{ [u]_r \mid u \in L \} \).

Prove that $\mathcal{L}(A) = L$. □
Isomorphism and Canonical Automata

Two automata \( A_i = \langle S_i, I_i, T_i, F_i \rangle \), \( i = 1, 2 \) are said to be \textbf{isomorphic} iff there exists a bijection \( h : S_1 \rightarrow S_2 \) such that, for all \( s, s' \in S_1 \) and for all \( \alpha \in \Sigma \) we have:

- \( s \in I_1 \iff h(s) \in I_2 \),
- \( (s, \alpha, s') \in T_1 \iff (h(s), \alpha, h(s')) \in T_2 \),
- \( s \in F_1 \iff h(s) \in F_2 \).

For the DFA class all minimal automata are isomorphic (why?)

For the NFA class there may be more non-isomorphic minimal automata.
**Pumping Lemma**

**Lemma 5 (Pumping)** Let $A = \langle S, I, T, F \rangle$ be a finite automaton with $\text{size}(A) = n$, and $w \in \mathcal{L}(A)$ be a word of length $|w| \geq n$. Then there exists three words $u, v, t \in \Sigma^*$ such that:

1. $|v| \geq 1,$
2. $w = uvt$ and,
3. for all $k \geq 0$, $uv^kt \in \mathcal{L}(A).$
Example

\[ L = \{ a^n b^n \mid n \in \mathbb{N} \} \] is not rational:

Suppose that there exists an automaton \( A \) with \( \text{size}(A) = N \), such that \( L = \mathcal{L}(A) \).

Consider the word \( a^N b^N \in L = \mathcal{L}(A) \).

There exists words \( u, v, w \) such that \( |v| \geq 1, uvw = a^N b^N \) and \( uv^k w \in L \) for all \( k \geq 1 \).

- \( v = a^m \), for some \( m \in \mathbb{N} \).
- \( v = a^m b^p \) for some \( m, p \in \mathbb{N} \).
- \( v = b^m \), for some \( m \in \mathbb{N} \).
Decidability

Given automata $A$ and $B$:

- **Emptiness** $\mathcal{L}(A) = \emptyset$ ?
- **Equality** $\mathcal{L}(A) = \mathcal{L}(B)$ ?
- **Infinity** $\|\mathcal{L}(A)\| < \infty$ ?
- **Universality** $\mathcal{L}(A) = \Sigma^*$ ?
Emptiness

Theorem 6  Let $A$ be an automaton with $\text{size}(A) = n$. If $\mathcal{L}(A) \neq \emptyset$, then there exists a word of length less than $n$ that is accepted by $A$.

Let $u$ be the shortest word in $\mathcal{L}(A)$.

If $|u| < n$ we are done.

If $|u| \geq n$, there exists $u_1, v, u_2 \in \Sigma^*$ such that $|v| > 1$ and $u_1vu_2 = u$.

Then $u_1u_2 \in \mathcal{L}(A)$ and $|u_1u_2| < |u_1vu_2|$, contradiction.
Everything is decidable

**Theorem 7** The emptiness, equality, infinity and universality problems are decidable for automata on finite words.

Use the Pumping Lemma to show decidability of emptiness.

Use the closure properties to show decidability of equality and universality.
Automata on Finite Words and WS1S
Let $\Sigma = \{a, b, \ldots\}$ be a finite alphabet.

Any finite word $w \in \Sigma^*$ induces the finite sets $p_a = \{p \mid w(p) = a\}$.

- $x \leq y$: $x$ is less than $y$,
- $S(x) = y$: $y$ is the successor of $x$,
- $p_a(x)$: $a$ occurs at position $x$ in $w$

Remember that $\leq$ and $S$ can be defined one from another (how?)
**Problem Statement**

Let $\mathcal{L}(\varphi) = \{ w \mid m_w \models \varphi \}$

A language $L \subseteq \Sigma^*$ is said to be WS1S-**definable** iff there exists a WS1S formula $\varphi$ such that $L = \mathcal{L}(\varphi)$.

1. Given $A$ build $\varphi_A$ such that $\mathcal{L}(A) = \mathcal{L}(\varphi)$

2. Given $\varphi$ build $A_\varphi$ such that $\mathcal{L}(A) = \mathcal{L}(\varphi)$

   The rational and WS1S-definable languages coincide
Coding of $\Sigma$

Let $m \in \mathbb{N}$ be the smallest number such that $\|\Sigma\| \leq 2^m$.

W.l.o.g. assume that $\Sigma = \{0, 1\}^m$, and let $X_1 \ldots X_p, x_{p+1}, \ldots x_m$

A word $w \in \Sigma^*$ induces an interpretation of $X_1 \ldots X_p, x_{p+1}, \ldots x_m$:

- $i \in I_w(X_j)$ iff the $j$-th element of $w_i$ is 1, and
- $I_w(x_j) = i$ iff $w_i$ has 1 on the $j$-th position and, for all $k \neq i$ $w_k$ has 0 on the $j$-th position.

We define $w \models \varphi(X_1, \ldots, X_p, x_{p+1}, \ldots, x_m)$. 
Example 2 Let \( \Sigma = \{a, b, c, d\} \), encoded as \( a = (00), b = (01), c = (10) \) and \( d = (11) \). Then the word \( abbaacdd \) induces the valuation \( X_1 = \{5, 6, 7\} \), \( X_2 = \{1, 2, 6, 7\} \). \( \square \)
From Automata to Formulae

Let \( A = \langle S, I, T, F \rangle \) with \( S = \{s_1, \ldots, s_p\} \), and \( \Sigma = \{0, 1\}^m \).

Build \( \Phi_A(X_1, \ldots, X_m) \) such that \( \forall w \in \Sigma^* . w \in \mathcal{L}(A) \iff w \models \Phi_A \)

Let \( a \in \{0, 1\}^m \). Let \( \Psi_a(x, X_1, \ldots, X_m) \) be the conjunction of:

- \( X_i(x) \) if the \( a_i = 1 \), and
- \( \neg X_i(x) \) otherwise.

For all \( w \in \Sigma^* \) we have \( w \models \forall x . \bigvee_{a \in \Sigma} \Psi_a(x, X) \)
Coding of $S$

Let $\{Y_0, \ldots, Y_p\}$ be set variables.

$Y_i$ is the set of all positions labeled by $A$ with state $s_i$ during some run

$$\Phi_S(Y_1, \ldots, Y_p) : \forall z . \bigvee_{1 \leq i \leq p} Y_i(z) \land \bigwedge_{1 \leq i < j \leq p} \neg \exists z . Y_i(z) \land Y_j(z)$$
Coding of $I$

Every run starts from an initial state:

$$\Phi_I(Y_1, \ldots, Y_p) : \exists x \forall y . x \leq y \land \bigvee_{s_i \in I} Y_i(x)$$
Coding of $T$

Consider the transition $s_i \xrightarrow{a} s_j$:

$$\Phi_T(X_1, \ldots, X_m, Y_1, \ldots, Y_p) : \forall x. x \neq S(x) \land Y_i(x) \land \Psi_a(x, X) \rightarrow \bigvee_{(s_i, a, s_j) \in T} Y_j(S(x))$$
The last state on the run is a final state:

\[ \Phi_F(Y_1, \ldots, Y_p) : \exists x \forall y . y \leq x \land \bigvee_{s_i \in F} Y_i(x) \]

\[ \Phi_A = \exists Y_1 \ldots \exists Y_p . \Phi_S \land \Phi_I \land \Phi_T \land \Phi_F \]
From Formulae to Automata

Let $\Phi(X_1, \ldots, X_p, x_{p+1}, \ldots, x_m)$ be a WS1S formula.

We build an automaton $A_\Phi$ such that $\mathcal{L}(A) = \mathcal{L}(\Phi)$.

Let $\Phi(X_1, X_2, x_3, x_4)$ be:

1. $X_1(x_3)$
2. $x_3 \leq x_4$
3. $X_1 = X_2$
From Formulae to Automata

$A_{\Phi}$ is built by induction on the structure of $\Phi$:

- for $\Phi = \phi_1 \land \phi_2$ we have $\mathcal{L}(A_{\Phi}) = \mathcal{L}(A_{\phi_1}) \cap \mathcal{L}(A_{\phi_2})$
- for $\Phi = \phi_1 \lor \phi_2$ we have $\mathcal{L}(A_{\Phi}) = \mathcal{L}(A_{\phi_1}) \cup \mathcal{L}(A_{\phi_2})$
- for $\Phi = \neg \phi$ we have $\mathcal{L}(A_{\Phi}) = \overline{\mathcal{L}(A_{\phi})}$
- for $\Phi = \exists X_i . \phi$, we have $\mathcal{L}(A_{\Phi}) = \text{pr}_i(\mathcal{L}(A_{\phi}))$. 
Consequences

Theorem 8  A language \( L \subseteq \Sigma^* \) is definable in WS1S iff it is rational.

Corollary 1  The SAT problem for WS1S is decidable.

Lemma 6  Any WS1S formula \( \phi(X_1, \ldots, X_m) \) is equivalent to an WS1S formula of the form \( \exists Y_1 \ldots \exists Y_p . \phi \), where \( \phi \) does not contain other set variables than \( X_1, \ldots, X_m, Y_1, \ldots, Y_p \).