Introduction to Logic and Automata Theory

Radu Iosif
Verimag/CNRS (Grenoble, France)
Ensuring Correctness of Hw/Sw Systems

• Uses logic to specify correctness properties, e.g.:
  – *the program never crashes*
  – *the program always terminates*
  – *every request to the server is eventually answered*
  – *the output of the tree balancing function is a tree, provided the input is also a tree ...*

• Given a logical specification, we can do either:
  – **VERIFICATION**: prove that a given system satisfies the specification
  – **SYNTHESIS**: build a system that satisfies the specification
Approaches to Verification

- **THEOREM PROVING**: reduce the verification problem to the satisfiability of a logical formula (entailment) and invoke an off-the-shelf theorem prover to solve the latter
  - Floyd-Hoare checking of **pre-, post-conditions** and **invariants**
  - Certification and Proof-Carrying Code

- **MODEL CHECKING**: enumerate the states of the system and check that the transition system satisfies the property
  - **explicit-state** model checking (SPIN)
  - **symbolic** model checking (SMV)

- **COMBINED METHODS**:
  - **static analysis** (ASTREE)
  - **predicate abstraction** (SLAM, BLAST)
Approaches to Synthesis

- **TREE AUTOMATA:**
  - starting point: logical specification
  - build word automaton from logic formula
  - transform into tree automaton
  - decide emptiness and build system from witness tree

- **CONTROL and GAME THEORY:**
  - starting point: incomplete/uncontrolled system with two types of freedom (system/environment choice) and an objective
  - the uncontrolled system is given as a game
  - controller/strategy tell how to achieve objective
Logic and Automata Connection

Given an automaton $A$, we build a logical formula $\varphi_A$ whose set of models is exactly the language of the automaton.

Given a logical formula $\varphi$, we build an automaton $A_\varphi$ that recognizes the set of all structures (models) in which $\varphi$ holds.

Assuming that $A_\varphi$ belongs to a well-behaved class of automata, we can tackle the following problems:

- **SATISFIABILITY**: $\varphi$ has a model if and only if $A_\varphi$ is not empty
- **MODEL CHECKING**: a given structure is a model of $\varphi$ if and only if it belongs to the language of $A_\varphi$
# Overview: Word and Tree Logics

<table>
<thead>
<tr>
<th></th>
<th>First Order Logic</th>
<th>⊆ Monadic Second Order Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>finite words</td>
<td>LTL, Star Free, Aperiodic Sets</td>
<td>Finite Automata</td>
</tr>
<tr>
<td>infinite words</td>
<td>LTL, Star Free, Aperiodic Sets</td>
<td>Büchi, Rabin Automata</td>
</tr>
<tr>
<td>finite trees</td>
<td>*</td>
<td>Tree Automata</td>
</tr>
<tr>
<td>infinite trees</td>
<td>*</td>
<td>Rabin Automata, Games</td>
</tr>
</tbody>
</table>
Overview: Integer Logics

Presburger Arithmetic $\subseteq \langle \mathbb{N}, +, V_p \rangle$

Semilinear Sets $\quad p$-automata
Preliminaries
**Words**

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

A *word* of length $n$ over $\Sigma$ is a sequence $w = a_0a_1 \ldots a_{n-1}$, where $a_i \in \Sigma$, for all $0 \leq i < n$. An *infinite word* is an infinite sequence of elements of $\Sigma$.

Equivalently, a word is a function $w : \{0, 1, \ldots, n-1\} \rightarrow \Sigma$. The *length* $n$ of the word $w$ is denoted by $|w|$. The *empty word* is denoted by $\epsilon$, i.e. $|\epsilon| = 0$.

$\Sigma^*$ ($\Sigma^\omega$) is the set of all finite (infinite) words over $\Sigma$, and $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$. We denote $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$.

The *concatenation* of two words $w$ and $u$ is denoted as $wu$. The *prefix* $u$ of $w$ is defined as $u \leq w$ iff there exists $v \in \Sigma^*$ such that $uv = w$. 
Trees

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

A **prefix-free** set $S \in \Sigma^*$ is such that for all $u, v \in S$, $u \neq v \Rightarrow u \not\leq v$ and $v \not\leq u$.

A **tree** over $\Sigma$ is a **partial function** $t : \mathbb{N}^* \mapsto \Sigma$ 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 iff it has an infinite path.
Ranked Trees

A ranked alphabet $\langle \Sigma, \# \rangle$ is a set of symbols together with a function $\# : \Sigma \rightarrow \mathbb{N}$. 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 $t : \mathbb{N}^* \rightarrow \Sigma$ that satisfies the following conditions:

- $dom(t)$ is a finite prefix-closed subset of $\mathbb{N}^*$, and

- for each $p \in dom(t)$, if $\#(t(p)) = n > 0$ then
  \[ \{i \mid pi \in dom(t)\} = \{1, \ldots, n\} \].

A symbol of arity zero is also called a constant. 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**: $\lor, \land, \rightarrow, \leftrightarrow, \neg, \bot, \forall, \exists$
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:

- ⊥ and ⊤ are formulae,
- 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 $\bullet \in \{\lor, \land, \to, \leftrightarrow\}$,
- nothing else is a formula.

The *language* of logic FOL is the set of formulae, denoted as $\mathcal{L}(\text{FOL})$. 
FOL Formulae

\[ x = y \]

\[ \forall x \forall y . \; x = y \iff y = x \]

\[ \exists x (\forall y . \; p(x,y)) \rightarrow q(x) \]

\[ \forall x . \; p(x) \rightarrow q(f(x)) \]

\[ \forall x \exists y . \; f(x) = y \land (\forall z . \; f(z) = y \rightarrow z = x) \]
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  $FV(\forall x . x = y \land x = z \to p(x)) = \{y, z\}$

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**,
- $\bar{f}_i : U^{\#(f_i)} \rightarrow U$, $i = 1, 2, \ldots$ are the **functions**,

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

**NB:** Every constant $c$ from the alphabet of FOL has a corresponding individual $\bar{c}$, but not vice versa.

The symbol $0$ has a corresponding number $\bar{0} \in \mathbb{N}$, and the function symbol $s$ has a corresponding function $x \mapsto x + 1$. The number $\bar{1} \in \mathbb{N}$ is denoted as $s(0)$, the number $\bar{2} \in \mathbb{N}$ as $s(s(0))$, etc.
**Semantics**

Let \( m = \langle U, p_1, p_2, \ldots, f_1, f_2, \ldots \rangle \) be a *structure*.

The interpretation of variables is a function:

\[
i : \{x, y, z, \ldots \} \rightarrow U
\]

The interpretation function is extended to terms \( t \), denoted as \( \iota(t) \in U \):

\[
\begin{align*}
\iota(c) & = \bar{c} \\
\iota(f(t_1, \ldots, t_n)) & = \bar{f}(\iota(t_1), \ldots, \iota(t_n))
\end{align*}
\]
Semantics

The meaning of a sentence \( \varphi \) in the structure \( m \) under the interpretation \( \iota \) is denoted as \( \models_\iota \varphi \in \{\text{true}, \text{false}\} \):

\[
\begin{align*}
\models_\iota \bot & = \text{false} \\
\models_\iota p(t_1, \ldots, t_n) & = \text{true} \text{ iff } \langle \iota(t_1), \ldots, \iota(t_n) \rangle \in \bar{p} \\
\models_\iota t_1 = t_2 & = \text{true} \text{ iff } \iota(t_1) = \iota(t_2) \\
\models_\iota \neg \varphi & = \text{true} \text{ iff } \models_\iota \varphi = \text{false} \\
\models_\iota \varphi \land \psi & = \text{true} \text{ iff } \models_\iota \varphi = \text{true} \text{ and } \models_\iota \psi = \text{true} \\
\models_\iota \exists x . \varphi & = \text{true} \text{ iff } \models_\iota \varphi[x \leftarrow u] = \text{true} \text{ for some } u \in U
\end{align*}
\]

where \( \iota[x \leftarrow u](y) = \iota(y) \) if \( x \neq y \) and \( \iota[x \leftarrow u](x) = u \).
Semantics

Derived meanings:

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

If \( FV(\varphi) = \emptyset \) we denote the meaning of \( \varphi \) in \( m \) by \( [\varphi]^m \) (the choice of \( \iota \) is irrelevant).

If \( [\varphi]^m = \text{true} \) we say that \( m \) is a \textit{model} of \( \varphi \), denoted as \( m \models \varphi \).

If \( m \models \varphi \) for all structures \( m \), we say that \( \varphi \) is \textit{valid}, denoted as \( \models \varphi \).

If \( \varphi \) has at least one model, we say that it is \textit{satisfiable}.

\textbf{Satisfiability}: Given \( \varphi \) is it satisfiable?

\textbf{Model Checking}: Given \( m \) and \( \varphi \), does \( m \models \varphi \)?
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:

- $\varphi_1 : \forall x \forall y . x \leq y \land y \leq x \leftrightarrow x = y$
- $\varphi_2 : \forall x \forall y \forall z . x \leq y \land y \leq z \rightarrow x \leq z$

Notice that $\models \varphi_1 \rightarrow \forall x . x \leq x$.

$m$ is a linearly ordered set if $m \models \varphi_1 \land \varphi_2 \land \varphi_3$, where:

- $\varphi_3 : \forall x \forall y . x \leq y \lor y \leq x$
Exercises

Exercise 1 Two problems $P$ and $Q$ are equivalent when a method for solving $P$ is also a method for solving $Q$, and vice versa. Show that satisfiability and validity of first-order sentences are equivalent problems. □

Exercise 2 Prove the validity of the following sentences:

\[
\forall x \forall y \forall z . x = y \land y = z \rightarrow x = z
\]

\[
(\exists x . \varphi \lor \psi) \leftrightarrow ((\exists x . \varphi) \lor (\exists x . \psi))
\]

\[
(\forall x . \varphi \land \psi) \leftrightarrow ((\forall x . \varphi) \land (\forall x . \psi))
\]

\[
(\exists x . \varphi \land \psi) \rightarrow ((\exists x . \varphi) \land (\exists x . \psi))
\]

\[
\neg((\exists x . \varphi) \land (\exists x . \psi)) \rightarrow (\exists x . \varphi \land \psi)
\]

\[
((\forall x . \varphi) \lor (\forall x . \psi)) \rightarrow (\forall x . \varphi \lor \psi)
\]

\[
\neg((\forall x . \varphi \lor \psi) \rightarrow ((\forall x . \varphi) \lor (\forall x . \psi)))
\]
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

$$\varphi = Q_1 x_1 Q_2 x_2 \ldots Q_n x_n \cdot \psi(x_1, x_2, \ldots, x_n)$$

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 \textit{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 \textit{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 a finite alphabet and $w : \{0, 1, \ldots, n - 1\} \to \Sigma$ be a finite word, i.e. $w = a_0 a_1 \ldots a_{n-1} \in \Sigma^*$.

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

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

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

Exercise 3  Write a FOL formula $S(x, y)$ which is valid for all positions $x, y \in \mathbb{N}$ such that $y = x + 1$. □

Exercise 4  Write a FOL sentence whose models are all words with $a$ on even positions and $b$ on odd positions. Next, (try to) write a FOL sentence whose models are all words with $a$ on even positions. □

Exercise 5  Write a FOL formula $\text{len}(x)$ that is satisfied by all words of length $x$. □

Exercise 6  Write a FOL sentence whose models are all finite words. □
Subword Formulae

Let \( w = a_0a_1 \ldots a_{n-1} \) be a finite word, and \( w(i, j) = a_i a_{i+1} \ldots a_{j-1} \) be a subword of \( w \), \( 0 \leq i < n \) and \( 0 \leq j \leq n, i < j \).

**Proposition 1** For each FOL statement \( \varphi \) there exists a formula \( \varphi(x, y) \) such that, for each \( w \in \Sigma^* \) and each \( 0 \leq i < j \leq |w| \):

\[
\begin{align*}
w(i, j) \models \varphi & \iff w \models \varphi(i, j) \\
(\neg \varphi)(x, y) & = \neg(\varphi(x, y)) \\
(\varphi \land \psi)(x, y) & = (\varphi(x, y)) \land (\psi(x, y)) \\
(\exists z. \varphi)(x, y) & = \exists z . x \leq z \land z < y \land \varphi(x, y)
\end{align*}
\]
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 $t : \mathbb{N}^* \mapsto \Sigma$ be a finite tree over $\Sigma$.

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

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

$m_{f(f(g,g),g)} = \langle \{\epsilon, 0, 1, 00, 01\}, p_f = \{\epsilon, 0\}, p_g = \{00, 01, 1\}, \preceq, \{s_n\}_{n \in \mathbb{N}} \rangle$. 
Examples

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

$$x \preceq y : x \leq y \lor \exists z . s_0(z) \leq x \land s_1(z) \leq y$$

**Exercise 7** A red-black tree is a tree in which all nodes are either red or black, such that the root is black, and each red node has only black children. Write a FOL sentence whose models are all red-black trees. ☐
FOL on Infinite Trees

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

The structure corresponding to \( t \) is \( m_t = \langle \mathbb{N}^*, \{\bar{p}_f\}_{f \in \Sigma}, \preceq, \{s_n\}_{n \in \mathbb{N}} \rangle \).
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 $\varphi \bullet \psi$, $\neg \varphi$, $\forall x . \varphi$, $\exists x . \varphi$, $\forall X . \varphi$ and $\exists X . \varphi$ are formulae, for $\bullet \in \{ \lor, \land, \rightarrow, \leftrightarrow \}$.

$X(t)$ is sometimes written $t \in X$. 


Examples

Universal set:

\[ \forall x . X(x) \]

\( X \subseteq Y \):

\[ \forall x . X(x) \rightarrow Y(x) \]

\( X \neq Y \):

\[ \exists x . (X(x) \land \neg Y(x)) \lor (\neg X(x) \land Y(x)) \]

\( X = \emptyset \):

\[ \forall x . \neg X(x) \]

Singleton set:

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

Let \( m = \langle U, \bar{p}_1, \bar{p}_2, \ldots, \bar{f}_1, \bar{f}_2, \ldots \rangle \) be a structure.

The interpretation of variables is a function:

\[
i: \{x, y, z, \ldots\} \cup \{X, Y, Z, \ldots\} \rightarrow U \cup 2^U
\]

such that:

- \( i(x) \in U \) for each individual variable \( x \)
- \( i(X) \in 2^U \) for each set variable \( X \)

\[
[\exists X . \varphi]_i^m = \text{true} \iff [\varphi]_{i[X \leftarrow S]}^m = \text{true} \text{ for some } S \subseteq U
\]
Example 2  The MSOL formula that characterizes all partitions \( \langle X, Y \rangle \) of \( Z \):

\[
\text{partition}(X, Y, Z) : (\forall x \forall y . X(x) \land Y(y) \rightarrow \neg x = y) \land (\forall x . Z(x) \leftrightarrow X(x) \lor Y(x))
\]
Let $\Sigma = \{a, b, \ldots\}$ be a finite alphabet. The alphabet of the sequential calculus is composed of:

- the function symbol $s$ denotes the successor,
- 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.

(W)eat indicates that quantification is over finite sets only.

**Q:** Let $m_{abbaab} = \langle\{0, \ldots, 5\}, \bar{p}_a = \{0, 3, 4\}, \bar{p}_b = \{1, 2, 5\}, \preceq\rangle$ be a finite word. How much is $s(5)$?
Examples

The order $x \leq y$ on positions is defined as:

- $\text{closed}(X) : \forall x . X(x) \rightarrow X(s(x))$
- $x \leq y : \forall X . X(x) \land \text{closed}(X) \rightarrow X(y)$

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

The first position is:

\[ \text{zero}(x) : \forall y . x \leq y \]

The set of even positions is defined by

\[ \text{even}(X) : \exists z . \text{zero}(z) \land \exists Y, Z . \text{pos}(Z) \land \text{partition}(X, Y, Z) \land \]
\[ \forall x, y . X(x) \land s(x) = y \rightarrow Y(y) \land \]
\[ \forall x, y . Y(x) \land s(x) = y \rightarrow Y(x) \land X(z) \]

The set of all words having \( a \)'s on even positions is the set of models of the sentence:

\[ \exists X . \text{even}(X) \land \forall x . X(x) \rightarrow p_a(x) \]
Exercise

Exercise 8  Write a $S1S$ formula whose models are exactly all infinite words starting with an even number of 0’s followed by an infinite number of 1’s. □
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.

In FOL on trees we had $\leq$ (prefix) instead of $s_i$. Why?
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 $\text{root}(x) : \forall y . x \leq y$. 
Exercise

Exercise 9  Define the set of binary trees \( t : \{0, 1\}^* \rightarrow \{a, b\} \) such that \( t(p) = a \) if \( p \) is of even length and \( t(p) = b \) if \( p \) is of odd length. \( \square \)

Exercise 10  Write a \( S\omega S \) formula \( \text{path}(X) \) that defines the set of all paths in a binary tree. \( \square \)

Exercise 11  Write a \( S\omega S \) sentence whose models are all finite trees. \( \square \)