

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** φ_A whose set of models is exactly the language of the automaton.

Given a **logical formula** φ , we build an **automaton** A_φ that recognizes the set of all structures (models) in which φ holds.

Assuming that A_φ belongs to a well-behaved class of automata, we can tackle the following problems:

- **SATISFIABILITY**: φ has a model if and only if A_φ is not empty
- **MODEL CHECKING**: a given structure is a model of φ if and only if it belongs to the language of A_φ

Overview: Word and Tree Logics

	First Order Logic	\subset Monadic Second Order Logic
finite words	LTL, Star Free, Aperiodic Sets	Finite Automata
infinite words	LTL, Star Free, Aperiodic Sets	Büchi, Rabin Automata
finite trees	*	Tree Automata
infinite trees	*	Rabin Automata, Games

Overview: Integer Logics

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

Semilinear Sets p -automata

(provided as additional material)

Preliminaries

Words

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

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

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

An infinite word is a function $w : \mathbb{N} \rightarrow \Sigma$.

Σ^* (Σ^ω) is the set of all finite (infinite) words over Σ , 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 . Note that $w \in \Sigma^*$, whereas $u \in \Sigma^\infty$. The *prefix* u of w is defined as $u \leq w$ iff there exists $v \in \Sigma^\infty$ such that $uv = w$.

Trees

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

A *prefix-free* set $S \subseteq \Sigma^*$ 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 Σ is a *partial function* $t : \mathbb{N}^* \mapsto \Sigma$ such that $\text{dom}(t)$ is a
prefix-closed set.

The *children* of a tree node $w \in \text{dom}(t)$ are all nodes $wn \in \text{dom}(t)$, such
that $n \in \mathbb{N}$. 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.

Trees (contd)

A *path* π is a set of nodes from $\text{dom}(t)$, such that:

1. the root belongs to the path i.e., $\epsilon \in \pi$,
2. for each node $p \in \pi$, exactly one of its children (if any) is on π .
3. for each node $pn \in \pi$, such that $n \in \mathbb{N}$, we have $p \in \pi$.

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 $\# : \Sigma \rightarrow \mathbb{N}$. For $f \in \Sigma$, the value $\#(f)$ is said to be the *arity* of f .

A *ranked tree* t over Σ is a partial function $t : \mathbb{N}^* \mapsto \Sigma$ that satisfies the following conditions:

- $\text{dom}(t)$ is a prefix-closed subset of \mathbb{N}^* , and
- for each $p \in \text{dom}(t)$, if $\#(t(p)) > 0$ then $\{i \mid pi \in \text{dom}(t)\} = \{1, \dots, \#(t(p))\}$.

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*: $p_1, p_2, \dots, =$
- *function symbols*: f_1, f_2, \dots
- *constant symbols*: c_1, c_2, \dots
- *first-order variables*: x, y, z, \dots
- *connectives*: $\vee, \wedge, \rightarrow, \leftrightarrow, \neg, \perp, \forall, \exists$

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, \dots, t_n are terms and f is a function symbol of arity $n > 0$, then $f(t_1, t_2, \dots, t_n)$ is a term,
- nothing else is a term.

A term with no variable is said to be a *ground term*.

Syntax

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

- \perp and \top are formulae,
- if t_1, t_2, \dots, t_n are terms and p is a predicate symbol of arity $n > 0$, then $p(t_1, t_2, \dots, t_n)$ is a formula,
- if t_1, t_2 are terms, then $t_1 = t_2$ is a formula,
- if φ and ψ are formulae, then $\varphi \bullet \psi$, $\neg\varphi$, $\forall x . \varphi$ and $\exists x . \varphi$ are formulae, for $\bullet \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$,
- nothing else is a formula.

An *atomic proposition* is any formula $p(t_1, \dots, t_n)$ or $t_1 = t_2$, where p is a predicate symbol and t_1, t_2, \dots, t_n are terms.

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

FOL Formulae

$$x = y$$

$$\forall x \forall y . x = y \leftrightarrow y = x$$

$$\forall x (\exists y . p(x, y)) \rightarrow q(x)$$

$$\forall x . p(x) \rightarrow q(f(x))$$

$$\forall x \exists y . f(x) = y \wedge (\forall z . f(z) = y \rightarrow z = x)$$

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 φ 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 φ . If $FV(\varphi) = \emptyset$ then φ is said to be a *sentence*.

Example 1 $FV(\forall x . x = y \wedge x = z \rightarrow p(x)) = \{y, z\} \square$

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

Semantics

A *structure* is a tuple $\mathfrak{m} = \langle U, \bar{p}_1, \bar{p}_2, \dots, \bar{f}_1, \bar{f}_2, \dots \rangle$, where:

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

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

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

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 $\mathfrak{m} = \langle U, \bar{p}_1, \bar{p}_2, \dots, \bar{f}_1, \bar{f}_2, \dots \rangle$ be a *structure*.

The *interpretation* of variables is a function:

$$\iota : \{x, y, z, \dots\} \rightarrow U$$

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

$$\begin{aligned} \iota(c) &= \bar{c} \\ \iota(f(t_1, \dots, t_n)) &= \bar{f}(\iota(t_1), \dots, \iota(t_n)) \end{aligned}$$

Semantics

The *meaning of a sentence φ in the structure \mathfrak{M} under the interpretation ι* is denoted as $\llbracket \varphi \rrbracket_{\iota}^{\mathfrak{M}} \in \{\text{true}, \text{false}\}$:

$$\llbracket \perp \rrbracket_{\iota}^{\mathfrak{M}} = \text{false}$$

$$\llbracket p(t_1, \dots, t_n) \rrbracket_{\iota}^{\mathfrak{M}} = \text{true} \quad \text{iff} \quad \langle \iota(t_1), \dots, \iota(t_n) \rangle \in \bar{p}$$

$$\llbracket t_1 = t_2 \rrbracket_{\iota}^{\mathfrak{M}} = \text{true} \quad \text{iff} \quad \iota(t_1) = \iota(t_2)$$

$$\llbracket \neg \varphi \rrbracket_{\iota}^{\mathfrak{M}} = \text{true} \quad \text{iff} \quad \llbracket \varphi \rrbracket_{\iota}^{\mathfrak{M}} = \text{false}$$

$$\llbracket \varphi \wedge \psi \rrbracket_{\iota}^{\mathfrak{M}} = \text{true} \quad \text{iff} \quad \llbracket \varphi \rrbracket_{\iota}^{\mathfrak{M}} = \llbracket \psi \rrbracket_{\iota}^{\mathfrak{M}} = \text{true}$$

$$\llbracket \exists x . \varphi \rrbracket_{\iota}^{\mathfrak{M}} = \text{true} \quad \text{iff} \quad \llbracket \varphi \rrbracket_{\iota[x \leftarrow u]}^{\mathfrak{M}} = \text{true} \quad \text{for some } u \in U$$

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

Semantics

Derived meanings:

$$\llbracket \varphi \vee \psi \rrbracket_{\iota}^{\mathfrak{m}} = \llbracket \neg(\neg\varphi \wedge \neg\psi) \rrbracket_{\iota}^{\mathfrak{m}}$$

$$\llbracket \varphi \rightarrow \psi \rrbracket_{\iota}^{\mathfrak{m}} = \llbracket \neg\varphi \vee \psi \rrbracket_{\iota}^{\mathfrak{m}}$$

$$\llbracket \varphi \leftrightarrow \psi \rrbracket_{\iota}^{\mathfrak{m}} = \llbracket (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi) \rrbracket_{\iota}^{\mathfrak{m}}$$

$$\llbracket \forall x . \varphi \rrbracket_{\iota}^{\mathfrak{m}} = \llbracket \neg\exists x . \neg\varphi \rrbracket_{\iota}^{\mathfrak{m}}$$

Decision Problems

If $FV(\varphi) = \emptyset$ we denote the meaning of φ in \mathfrak{m} by $\llbracket \varphi \rrbracket^{\mathfrak{m}}$ (the choice of ι is irrelevant)

If $\llbracket \varphi \rrbracket^{\mathfrak{m}} = \text{true}$ we say that \mathfrak{m} is a *model* of φ , denoted as $\mathfrak{m} \models \varphi$.

If $\mathfrak{m} \models \varphi$ for all structures \mathfrak{m} , we say that φ is *valid*, denoted as $\models \varphi$.

If φ has at least one model, we say that it is *satisfiable*.

Satisfiability: Given φ is it satisfiable?

Model Checking: Given \mathfrak{m} and φ , does $\mathfrak{m} \models \varphi$?

Examples

Let \leq be a binary predicate symbol, and $\mathfrak{M} = \langle U, \bar{\leq} \rangle$ be a structure. \mathfrak{M} is a **partially ordered set** if $\mathfrak{M} \models \varphi_1 \wedge \varphi_2$, where:

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

$$\varphi_2 : \forall x \forall y \forall z . x \leq y \wedge y \leq z \rightarrow x \leq z$$

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

\mathfrak{M} is a **linearly ordered set** if $\mathfrak{M} \models \varphi_1 \wedge \varphi_2 \wedge \varphi_3$, where:

$$\varphi_3 : \forall x \forall y . x \leq y \vee y \leq x$$

Exercises

Exercise 1 *Two problems P and Q are equivalent when a method for solving P is also a method for solving Q , and viceversa. Show that satisfiability and validity of first-order sentences are equivalent problems. \square*

Exercise 2 *Prove the validity of the following sentences:*

$$\forall x \forall y \forall z . x = y \wedge y = z \rightarrow x = z$$

$$(\exists x . \varphi \vee \psi) \leftrightarrow ((\exists x . \varphi) \vee (\exists x . \psi))$$

$$(\forall x . \varphi \wedge \psi) \leftrightarrow ((\forall x . \varphi) \wedge (\forall x . \psi))$$

$$(\exists x . \varphi \wedge \psi) \rightarrow ((\exists x . \varphi) \wedge (\exists x . \psi))$$

$$\neg(((\exists x . \varphi) \wedge (\exists x . \psi)) \rightarrow (\exists x . \varphi \wedge \psi))$$

$$((\forall x . \varphi) \vee (\forall x . \psi)) \rightarrow (\forall x . \varphi \vee \psi)$$

$$\neg((\forall x . \varphi \vee \psi) \rightarrow ((\forall x . \varphi) \vee (\forall x . \psi)))$$

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

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

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

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 λ_{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 λ_{ij} are either atomic propositions or negations of atomic propositions.

FOL on Finite Words

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

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

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

$$\mathfrak{m}_{abbaab} = \langle \{0, \dots, 5\}, \bar{p}_a = \{0, 3, 4\}, \bar{p}_b = \{1, 2, 5\}, \bar{\leq} \rangle$$

Exercises

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

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. \square

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

Exercise 6 Write a FOL sentence whose models are all finite words. \square

FOL on Infinite Words

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

The structure corresponding to w is $\mathfrak{m}_w = \langle \mathbb{N}, \{\bar{p}_a\}_{a \in \Sigma}, \bar{\leq} \rangle$.

$$\mathfrak{m}_{(ab)^\omega} = \langle \mathbb{N}, \bar{p}_a = \{2k \mid k \in \mathbb{N}\}, \bar{p}_b = \{2k + 1 \mid k \in \mathbb{N}\}, \bar{\leq} \rangle$$

FOL on Finite Trees

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

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

- $\bar{p}_f = \{p \in \text{dom}(t) \mid t(p) = f\}$,
- \preceq is the **prefix order** on \mathbb{N}^* ,
- $s_n(p) = \begin{cases} pn, & \text{if } pn \in \text{dom}(t) \\ p, & \text{otherwise} \end{cases}$ for all $n \in \mathbb{N}$, is the **n -th successor**.

Examples

$\mathfrak{m}_{f(f(g,g),g)} = \langle \{\epsilon, 0, 1, 00, 01, 10, 11\}, p_f = \{\epsilon, 0, 1\}, p_g = \{00, 01, 10, 11\}, \bar{\leq}, \{s_0, s_1\} \rangle$, where:

- $s_i(p) = pi$, for all $p \in \{\epsilon, 0, 1\}$ and $i \in \{0, 1\}$,
- $s_0(00) = s_1(00) = 00$, $s_0(01) = s_1(01) = 01$, $s_0(10) = s_1(10) = 10$ and $s_0(11) = s_1(11) = 11$.

The *lexicographic order* on $\{0, 1\}^*$ is defined as follows:

$$x \preceq_{lex} y \stackrel{\text{def}}{=} x \preceq y \vee \exists z . s_0(z) \preceq x \wedge s_1(z) \preceq y$$

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. \square

FOL on Infinite Trees

Let $t : \mathbb{N}^* \mapsto \Sigma$ be an **infinite tree** over Σ .

The structure corresponding to t is $\mathfrak{m}_t = \langle \mathbb{N}^*, \{\bar{p}_f\}_{f \in \Sigma}, \preceq, \{s_n\}_{n \in \mathbb{N}} \rangle$, where:

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

Exercise 8 Given a (possibly infinite) set $\mathcal{T} = \{t_1, t_2, \dots\}$ of finite or infinite trees, of finite or infinite branching degrees, represent each tree $t_i \in \mathcal{T}$ as an infinite binary tree $\bar{t}_i : \{0, 1\}^* \rightarrow \Sigma$. \square

Monadic Second Order Logic

Syntax

The alphabet of MSOL consists of:

- all first-order symbols
- *set variables*: X, Y, Z, \dots

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 φ and ψ are formulae, then $\varphi \bullet \psi$, $\neg\varphi$, $\forall x . \varphi$, $\exists x . \varphi$, $\forall X . \varphi$ and $\exists X . \varphi$ are formulae, for $\bullet \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$.

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

Examples

Universal set:

$$\forall x . X(x)$$

$X \subseteq Y$:

$$\forall x . X(x) \rightarrow Y(x)$$

$X \neq Y$:

$$\exists x . (X(x) \wedge \neg Y(x)) \vee (\neg X(x) \wedge Y(x))$$

$X = \emptyset$:

$$\forall x . \neg X(x)$$

Singleton set:

$$\forall Y . ((\forall x . Y(x) \rightarrow X(x)) \wedge \exists x . X(x) \wedge \neg Y(x)) \rightarrow \forall x . \neg Y(x)$$

Semantics

Let $\mathfrak{m} = \langle U, \bar{p}_1, \bar{p}_2, \dots, \bar{f}_1, \bar{f}_2, \dots \rangle$ be a *structure*.

The *interpretation* of variables is a function:

$$\iota : \{x, y, z, \dots\} \cup \{X, Y, Z, \dots\} \rightarrow U \cup 2^U$$

such that:

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

$$\llbracket \exists X . \varphi \rrbracket_{\iota}^{\mathfrak{m}} = \text{true} \quad \text{iff} \quad \llbracket \varphi \rrbracket_{\iota[X \leftarrow S]}^{\mathfrak{m}} = \text{true} \quad \text{for some } S \subseteq U$$

MSOL Example

Example 2 *The MSOL formula that characterizes all partitions $\langle X, Y \rangle$ of Z :*

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

□

MSOL on Words: (W)S1S

Let $\Sigma = \{a, b, \dots\}$ 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)eak indicates that quantification is over finite sets only.

Example 3 Q: Let $\mathfrak{m}_{abbaab} = \langle \{0, \dots, 5\}, \bar{p}_a = \{0, 3, 4\}, \bar{p}_b = \{1, 2, 5\}, \bar{s} \rangle$ be a finite word, where $\bar{s}(n) = n + 1$, for $n = 0, \dots, 4$ and $\bar{s}(5) = 5$. \square

Examples

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

- $closed(X) : \forall x . X(x) \rightarrow X(s(x))$
- $x \leq y : \forall X . X(x) \wedge closed(X) \rightarrow X(y)$

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

Examples

The first position is: $zero(x) : \forall y . x \leq y$

The set of even positions is defined by

$$\begin{aligned} even(X) : & \exists z . zero(z) \wedge X(z) \wedge \\ & \exists Y \exists Z . pos(Z) \wedge partition(X, Y, Z) \wedge \\ & \forall x \forall y . X(x) \wedge \neg s(x) = x \rightarrow Y(s(x)) \wedge \\ & \forall x \forall y . Y(x) \wedge \neg s(x) = x \rightarrow X(s(x)) \end{aligned}$$

The set of all words having a 's on even positions is the set of models of the sentence: $\exists X . even(X) \wedge \forall x . X(x) \rightarrow p_a(x)$

Exercise 9 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. \square

MSOL on Trees: (W)SkS

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

- the function symbols $\{s_i \mid i = 1, \dots, k\}$, where $s_i(x)$ denotes the *i-th successor* of x ; if we allow $\{s_i \mid i \in \mathbb{N}\}$, the logic is called (W)S ω S,
- the predicate symbols $\{p_a \mid a \in \Sigma\}$; 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 $closed(X) : \forall x . X(x) \rightarrow X(s_0(x)) \wedge X(s_1(x))$ denotes the fact that X is a **downward-closed** set.
- The **prefix ordering** on tree positions is defined by $x \leq y : \forall X . closed(X) \wedge X(x) \rightarrow X(y)$.
- The **root** of a tree is defined by $root(x) : \forall y . x \leq y$.

Exercise

Exercise 10 Define the set of binary trees $t : \{0, 1\}^* \rightarrow \{a, b\}$ such that $t(p) = a$ if p is of even length. \square

Exercise 11 Write a S2S formula $\text{path}(X)$ that defines the set of all paths in a binary tree. \square

Exercise 12 Write a S2S sentence whose models are all finite trees. \square