Monadic Second order logic on trees
(n-ary ordered) Tree domain $T$ is a subset of $\mathbb{F}_0, 1, 2, \ldots, n-1 \mathbb{F}_n^*$ that is
- prefix closed.
- If $u \in \mathbb{F}_0, \ldots, n-1 \mathbb{F}_n^*$, $i, j \in \mathbb{F}_0, \ldots, n-1 \mathbb{F}_n$ and $i < j$ then $u_j \in T$.

$\Sigma$-labeled n-ary trees are $T = (\text{dom}_T, \text{lab}_T)$ such that $\text{dom}_T$ is a tree domain and $\text{lab}_T : \text{dom}_T \to \Sigma$ is a labeling function.

Tree Automaton $A = (Q, \Sigma, S, F)$
- $S = \bigcup_{i=0}^{n} S_i$
- $S_i : Q^i \times \Sigma \to 2^Q$

Process labeled trees starting at the leaf and going up the tree until it reaches the root.

An input tree $\tilde{t}$ is accepted if $A$ has a run where the root is labeled by a final state.

Tree Regular Languages A set of $\Sigma$-labeled (n-ary) trees such that the set is recognized by some tree automaton.
Properties of Tree Regular Languages

- Deterministic tree automata is as powerful as a non-deterministic tree automata.

- Tree regular languages are closed under union, intersection and complementation.

- Other classical closure properties of word regular languages extend to tree languages.

- There are non-regular tree languages.

\[ L = \{ A(t_1, t_2) \mid t_1 = t_2 \} \]

This tree with root labeled `A` and two children which are trees `t_1` and `t_2` are isomorphic.

Proof

Suppose \( L \) is recognized by a tree automaton \( B \) with \( k \) states. There must be two trees \( t_1 \) and \( t_2 \) such that in the run of \( B \) on \( t_1 \), and \( t_2 \), the state labeling the root is the same.

\[ \vdash A(t_1, t_2) \in L \]
\( A(t_1, t_1) \in L \) (and \( A(t_2, t_2) \rightarrow \))

but \( A(t_1, t_2) \notin L \).

The run of \( B \) on \( A(t_1, t_1) \) and \( A(t_1, t_2) \)

will be in the same state at the

root.

**Proposition** Given a tree automaton \( A \) the problem of determining if \( L(A) = \emptyset \) and
the problem of determining if \( L(A) \) is

"universal" are decidable.

If \( \mathcal{A} \) is deterministic decidable in linear
time.

---

**\( \Sigma \)-labeled Trees as structures**

**signature**

\[ (\prec, \bar{S}_0, \bar{S}_1, \ldots, \bar{S}_{n+1}, \bar{\Sigma}, \bar{a}, \bar{\Sigma}, \bar{x}) \]

**tree as a structure**

\[ t = (\text{dom}_t, \text{lab}_t) \]

is a structure \( Y = (T, \prec^T, S_i^T, Q_0^T) \)

- \( T = \text{dom}_t \)
- \((u, v) \in \prec^T \) if \( u \) is a strict
  prefix of \( v \)
- \( S_i = \{ (u, vi) \mid u, vi \in T^3 \} \)
- \( \text{dom}_t = \{ \varepsilon, 0, 1, 10, 100, 101 \} \)
$Q_a = \exists u \in T \mid \text{lab}_b(u) = a^3$.

$T = \exists \alpha, 0, 1, 10, 100, 1000 \exists \gamma$

$\preceq = \exists (\alpha, 0), (\alpha, 1), (\alpha, 10) \ldots \exists$

$S_0 = \exists (\alpha, 0), (1, 10), (10, 100) \exists$

$S_1 = \exists (\alpha, 1), (10, 101) \exists$

$Q_\wedge = \exists \alpha, 10 \exists \gamma \exists \gamma_2 \ldots$

**Monad Logic on Trees.**

Second order logic formulas over the signature $(\preceq, S_0, \ldots, S_n, \exists \alpha \beta \gamma \exists \alpha \beta \gamma)$

such that all relational variables have arity 1.

Terms: $t ::= x$ — variables.

Formulas: $\phi ::= t_1 = t_2 \mid \preceq(t_1, t_2) \mid S_i(t_1, t_2)$

$Q_{\wedge} \mid \exists t$

$\neg \phi \mid \phi \lor \phi \rightarrow$ relational variables

$\exists x. \phi \mid \exists X \phi \rightarrow$ arity 1

For a sentence $\phi$

$[\phi] = \exists T \mid T \vdash \phi \exists

$\text{Trees defined by } \phi$. 

$\rightarrow$
Drez, Thatcher-Wright A tree language

$L$ is regular iff it is definable in MSO
i.e $\exists$ MSO sentence $\varphi$ s.t.

$$L = \{ c \in \mathbb{N} \}$$

Proof \((\Rightarrow)\) Let $L$ that is recognized by a tree automaton $A = (\mathcal{E}_0, \cdots, \mathcal{E}_{k-1}, \preceq, \delta, \bullet, F)$. The sentence defining $L$ is going say that the automaton $A$ has an accepting run on input tree.

Run is a tree labeled by $\mathcal{E}_0, \cdots, \mathcal{E}_{k-1}$

For each state $2$ we will identify the vertices that have label $1$ in the accepting run.

$$\varphi_A = \exists x_0 \exists x_1 \cdots \exists x_{k-1}$$

$$\forall x \left( \bigwedge_{i=0}^{k-1} \bigwedge_{i \neq j} (x_i \land x_j) \right)$$

Every vertex has a unique state label.

$$\forall x \left( \text{root}(x) \rightarrow \bigvee_{i \in F} x_i \right)$$

$\text{root}$ is labeled by a final state.
\[ \text{root}(x) = \forall y \wedge \bigwedge_{i=0}^{n-1} \forall x \forall y_0 \ldots \forall y_{i-1} \]

\[ \text{children}_i(x, y_0 \ldots y_{i-1}) \rightarrow \]

\[ \bigvee \left( \exists a(x) \wedge \bigwedge_{j=0}^{i-1} x_{p_j} y_j \wedge x_i \alpha \right) \]

\[ \text{pes}(p_0 \ldots p_{i-1}, \alpha) \]

Labeling of vertices by states is consistent with the transition \( \alpha \)

defined.

This is a formula of the form

\[ \exists X \forall y \varphi. \]