First Order and Chain Definability of Regular Tree Languages

Igor Walukiewicz (LaBRI); Mikołaj Bojanczyk (Warszawa)
Summary

Quick reminder of logic and languages
Summary

- Quick reminder of logic and languages
- Overview of FOL definable word languages
Summary

- Quick reminder of logic and languages
- Overview of FOL definable word languages
- FOL definable tree languages and some characterisations
Summary

- Quick reminder of logic and languages
- Overview of FOL definable word languages
- FOL definable tree languages and some characterisations
- Chain logic and some conjectures
Summary

- Quick reminder of logic and languages
- Overview of FOL definable word languages
- FOL definable tree languages and some characterisations
- Chain logic and some conjectures
- Conclusion
Let $\Sigma$ be an alphabet and $w = a_0 \ldots a_n$ a word over $\Sigma$. This word is represented as a relational structure

$$w = (\text{dom}(w), S^w, <^w, (Q^w_a)_{a \in \Sigma})$$

called the \textit{word model} for $w$, where $\text{dom}(w) = \{0, \ldots, n\}$, $S^w$ is the successor relation on $\text{dom}(w)$, $<^w$ is the natural order and $Q^w_a = \{i : a_i = a\}$. 
MSOL definability

A language $L \subseteq \Sigma^*$ is **MSOL definable** iff there exists an MSOL formula $\phi_L$ such that

$$w \in L \iff w \models \phi_L$$
MSOL definability

A language $L \subseteq \Sigma^*$ is **MSOL definable** iff there exists an MSOL formula $\phi_L$ such that

$$w \in L \iff w \models \phi_L$$

**Thm:** A language is MSOL definable iff it is regular
A language $L \subseteq \Sigma^*$ is \textit{FOL definable} iff there exists a FOL formula $\phi_L$ such that

$$w \in L \iff w \models \phi_L$$
FOL definability

A language $L \subseteq \Sigma^*$ is **FOL definable** iff there exists a FOL formula $\phi_L$ such that

$$w \in L \iff w \models \phi_L$$

The language $(ab)^*$ is FOL definable using the formula:

$$\forall x.[Q_a(x) \iff \exists y.(S(x,y) \land Q_b(y))]$$
FOL definability

A language $L \subseteq \Sigma^*$ is FOL definable iff there exists a FOL formula $\phi_L$ such that

$$w \in L \iff w \models \phi_L$$

The language $(ab)^*$ is FOL definable using the formula:

$$\forall x. [Q_a(x) \iff \exists y.(S(x,y) \land Q_b(y))]$$

The language $(aa)^*$ is not FOL definable
FOL definability criteria

Some characterisations of FOL definable word languages:
FOL definability criteria

Some characterisations of FOL definable word languages:

1. $L$ is star-free, that is defined by a regular expression using concatenation, sum and complementation. (McNaughton and Papert 71)
FOL definability criteria

Some characterisations of FOL definable word languages:

1. $L$ is star-free, that is defined by a regular expression using concatenation, sum and complementation. (McNaughton and Papert 71)

2. The syntactic semigroup of $L$ contains no nontrivial subgroup (Schützenberger 65).
FOL definability criteria

Some characterisations of FOL definable word languages:

1. \( L \) is star-free, that is defined by a regular expression using concatenation, sum and complementation. (McNaughton and Papert 71)

2. The syntactic semigroup of \( L \) contains no nontrivial subgroup (Schützenberger 65).

3. There is some \( n \in \mathbb{N} \) such that for all \( v, u, w \in \Sigma^* \)

\[ v(u^n)w \in L \iff v(u^{n+1})w \in L \]
FOL definability criteria

Some characterisations of FOL definable word languages:

1. $L$ is star-free, that is defined by a regular expression using concatenation, sum and complementation. (McNaughton and Papert 71)

2. The syntactic semigroup of $L$ contains no nontrivial subgroup (Schützenberger 65).

3. There is some $n \in \mathbb{N}$ such that for all $v, u, w \in \Sigma^*$

$$v(u^n)w \in L \iff v(u^{n+1})w \in L$$

4. $L$ is expressible in LTL (Kamp 68)
FOL definability criteria

Some characterisations of FOL definable word languages:

1. $L$ is star-free, that is defined by a regular expression using concatenation, sum and complementation. (McNaughton and Papert 71)

2. The syntactic semigroup of $L$ contains no nontrivial subgroup (Schutzenberger 65).

3. There is some $n \in \mathbb{N}$ such that for all $v, u, w \in \Sigma^*$

   $$v(u^n)w \in L \iff v(u^{n+1})w \in L$$

4. $L$ is expressible in LTL (Kamp 68)

**Cor:**[of 2,3] It is decidable whether a given regular language is FOL definable.
The tree case

For a finite binary tree $t$ a similar structure $\mathfrak{t}$ is considered:

$$\mathfrak{t} = (\text{dom}(t), S_0^t, S_1^t, <^t, (Q_a^t)_{a \in \Sigma})$$

where $\text{dom}(t) \subseteq \{0, 1\}^*$ is the set of nodes of the tree, $S_i^t$ denotes the $i$-th successor relation

$$S_i^t = \{(v, v \cdot i) : v, v \cdot i \in \text{dom}(t)\}$$

and $<^t, Q_a^t$ are defined as in the word case.
Thm: [Thatcher and Wright, Rabin] MSOL = regular.
**MSOL and FOL tree languages**

**Thm:** [Thatcher and Wright, Rabin] MSOL=regular.

1. The tree contains an odd number of nodes (MSOL)

\[
\exists X. \forall x. [\text{root}(x) \lor \text{leaf}(x)] \Rightarrow X(x) \land \\
(\forall x, x_0, x_1. [S_0(x, x_0) \land S_1(x, x_1)] \Rightarrow [X(x) \Leftrightarrow \neg(X(x_0) \Leftrightarrow X(x_1))])
\]
**MSOL and FOL tree languages**

**Thm:** [Thatcher and Wright, Rabin] MSOL=regular.

1. The tree contains an odd number of nodes (MSOL)

\[ \exists X. \forall x. [\text{root}(x) \lor \text{leaf}(x)] \Rightarrow X(x) \land (\forall x, x_0, x_1. [S_0(x, x_0) \land S_1(x, x_1)] \Rightarrow [X(x) \Leftrightarrow \neg(X(x_0) \Leftrightarrow X(x_1))]) \]

2. There exist two nodes labelled by \( a \) (FOL)

\[ \exists x, y. x \neq y \land Q_a(x) \land Q_a(y) \]
MSOL and FOL tree languages

**Thm:** [Thatcher and Wright, Rabin] MSOL=regular.

1. The tree contains an odd number of nodes (MSOL)

$$\exists X. \forall x. [\text{root}(x) \lor \text{leaf}(x)] \Rightarrow X(x) \land$$

$$(\forall x, x_0, x_1. [S_0(x, x_0) \land S_1(x, x_1)] \Rightarrow [X(x) \iff \neg (X(x_0) \iff X(x_1))]]$$

2. There exist two nodes labelled by $a$ (FOL)

$$\exists x, y. x \neq y \land Q_a(x) \land Q_a(y)$$

**Fact:** The property (1) is not FOL definable
Main question

Our unattained goal is two answer the question:

Given a regular tree language $L$ decide whether $L$ is FOL definable.
CTL* formulas over the alphabet $\Sigma = \{a_0, \ldots, a_n\}$ are defined by the following grammar:

$$F := \exists F \mid F\lor F \mid F \land F \mid \neg F \mid a_0 \mid \ldots \mid a_n$$
CTL*

CTL* formulas over the alphabet $\Sigma = \{a_0, \ldots, a_n\}$ are defined by the following grammar:

$$F := \exists F | F \cup F | F \land F | \neg F | a_0 | \ldots | a_n$$

Each CTL* formula $\psi$ is translated to a two-variable FOL formula $[\psi](x, y)$:

1. $[a_i](x, y) = Q_{a_i}(x)$
2. $[\psi \land \varphi](x, y) = [\psi](x, y) \land [\varphi](x, y)$
3. $[\neg \psi](x, y) = \neg[\psi](x, y)$
4. $[\psi \cup \varphi](x, y) = \exists z \leq y. [[\varphi](z, y) \land \forall z' \in (x; z).[\psi](z', z))]$
5. $[\exists \psi](x, y) = \exists y.[[\psi](x, y)]$
CTL* = FOL

Thm: CTL* = FOL, both on finite and infinite trees.
CTL* = FOL

Thm: CTL* = FOL, both on finite and infinite trees.

\[ \exists x. Q_c(x) \land \forall y < x. \exists z > y. (Q_a(z) \land \forall (x' \in [y; z]). Q_b(x')) \]

\[ \psi \cup^* \varphi := \psi \land (\psi \cup \varphi) \]

\[ \exists[(\exists b \cup^* a) \cup^* c] \]
\[ \exists \left( \exists b U^* a \right) U^* c \]
\( \exists[(\exists b U^* a) U^* c] \)
\[ \exists[(\exists b U^* a) U^* c] \]
\[ \exists (\exists b^* a)U^* c \]
Word-sum automata

Consider a deterministic word automaton $\mathcal{A} = \langle Q, q_0, \delta \rangle$ over the alphabet $\Sigma \times \{0,1\}$. Let $Q \cdot (a, i) = \{\delta(q, (a, i)) : q \in Q\}$. The automaton $\mathcal{A}_{ws} = \langle P(Q), \{q_0\}, \delta' \rangle$ is a automaton over $\Sigma$-labelled trees whose transition function $\delta'$ is defined as follows:

$$Q_0 \cdot (a, 0) \cup Q_1 \cdot (a, 1)$$

\[
\begin{array}{c}
\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ a \\
\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ / \ \\
\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ / \ \\
\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ / \ \\
Q_0 \quad Q_1
\end{array}
\]
Word-sum automata

Consider a deterministic word automaton $\mathcal{A} = \langle Q, q_0, \delta \rangle$ over the alphabet $\Sigma \times \{0, 1\}$. Let $Q \cdot (a, i) = \{\delta(q, (a, i)) : q \in Q\}$. The automaton $\mathcal{A}_{ws} = \langle P(Q), \{q_0\}, \delta' \rangle$ is a automaton over $\Sigma$-labelled trees whose transition function $\delta'$ is defined as follows:

$$Q_0 \cdot (a, 0) \cup Q_1 \cdot (a, 1)$$

Df: A tree automaton $\mathcal{A}$ is a word-sum automaton iff $\mathcal{A} = \mathcal{A}_{ws}'$ for some word automaton $\mathcal{A}'$. The automaton $\mathcal{A}$ is an aperiodic word-sum automaton if $\mathcal{A}'$ is aperiodic.
Word-sum automata, continued

For a tree language $L$, the following are equivalent:

- $L$ is definable by some word-sum automaton.
Word-sum automata, continued

For a tree language $L$, the following are equivalent:

- $L$ is definable by some word-sum automaton.
- $L$ is a boolean combination of deterministic top-bottom automata.
Word-sum automata, continued

For a tree language $L$, the following are equivalent:

- $L$ is definable by some word-sum automaton.
- $L$ is a boolean combination of deterministic top-bottom automata
- $L$ admits a certain slicing characterisation
For a tree language $L$, the following are equivalent:

- $L$ is definable by some (aperiodic) word-sum automaton.
- $L$ is a boolean combination of deterministic top-bottom (aperiodic) automata
- $L$ admits a certain slicing (aperiodic) characterisation
For a tree language $L$, the following are equivalent:

1. $L$ is definable by some (aperiodic) word-sum automaton.
2. $L$ is a boolean combination of deterministic top-bottom (aperiodic) automata.
3. $L$ admits a certain slicing (aperiodic) characterisation.

**Fact:** Aperiodic word-sum automata recognize precisely $\text{CTL}^*$ formulas of $\exists$-depth 1.
Word-sum automata, continued

For a tree language $L$, the following are equivalent:

- $L$ is definable by some (aperiodic) word-sum automaton.
- $L$ is a boolean combination of deterministic top-bottom (aperiodic) automata
- $L$ admits a certain slicing (aperiodic) characterisation

**Fact:** Aperiodic word-sum automata recognize precisely $\text{CTL}^*$ formulas of $\exists$-depth 1.

**Thm:** It is decidable whether a given language is word-sum definable.
Let $\mathcal{A} = \langle Q, q_s, \delta \rangle$ be an automaton over $\Sigma$ labelled trees and $\mathcal{A}' = \langle Q', q'_s, \delta' \rangle$ an automaton over $\Sigma \times Q$ labelled trees. Assume that both are bottom-up deterministic.
Wreath product

Let $\mathcal{A} = \langle Q, q_s, \delta \rangle$ be an automaton over $\Sigma$ labelled trees and $\mathcal{A}' = \langle Q', q'_s, \delta' \rangle$ an automaton over $\Sigma \times Q$ labelled trees. Assume that both are bottom-up deterministic.

**Df:** The *wreath* product of $\mathcal{A}'$ and $\mathcal{A}$ is the automaton $\mathcal{A}' \circ \mathcal{A} = \langle Q \times Q', (q_s, q'_s), \delta_\circ \rangle$ over $\Sigma$ labelled trees whose transition function is defined as follows:

$$\delta_\circ(((q_0, q'_0), a, (q_1, q'_1)) = (q, q')$$

where $q = \delta(q_0, q_1)$ and $q' = \delta'(q'_0, (a, q), q'_1)$.
Another characterisation

**Thm:** A language is FOL definable iff it is recognized by a wreath product of aperiodic word-sum languages.
Another characterisation

**Thm:** A language is FOL definable iff it is recognized by a wreath product of aperiodic word-sum languages

Since wreath product can simulate boolean combinations we also have:

**Thm:** A language is FOL definable iff it is recognized by a wreath product of aperiodic top-bottom deterministic languages
Another characterisation

**Thm:** A language is FOL definable iff it is recognized by a wreath product of aperiodic word-sum languages.

Since wreath product can simulate boolean combinations we also have:
**Thm:** A language is FOL definable iff it is recognized by a wreath product of aperiodic top-bottom deterministic languages.

**Question:** What if the word-sum languages are not aperiodic?
Another characterisation

**Thm:** A language is FOL definable iff it is recognized by a wreath product of aperiodic word-sum languages.

Since wreath product can simulate boolean combinations we also have:
**Thm:** A language is FOL definable iff it is recognized by a wreath product of aperiodic top-bottom deterministic languages.

**Question:** What if the word-sum languages are not aperiodic?

**Thm:** A language is chain definable iff it is recognized by a wreath product of word-sum languages.
**Chain logic**

**Df:** A set of tree vertices $C$ is a *chain* iff it is totally ordered by the relation $\leq$.

*Chain logic* (CL) has the same syntax as monadic second order logic, but the semantics for the monadic quantifier $\exists$ are different:

$$ t \models \exists X . \psi \iff \text{there is a chain } C \text{ such that } t[X := C] \models \psi $$
**Chain logic**

**Df:** A set of tree vertices $C$ is a *chain* iff it is totally ordered by the relation $\leq$.

*Chain logic* (CL) has the same syntax as monadic second order logic, but the semantics for the monadic quantifier $\exists$ are different:

$$t \models \exists X.\psi \text{ iff there is a chain } C \text{ such that } t[X := C] \models \psi$$

- Obviously $\text{FOL} \subseteq \text{CL} \subseteq \text{MSOL}$. 

---

First Order and Chain Definability of Regular Tree Languages – p.17/30
**Chain logic**

**Df:** A set of tree vertices \( C \) is a *chain* iff it is totally ordered by the relation \( \leq \).

*Chain logic* (CL) has the same syntax as monadic second order logic, but the semantics for the monadic quantifier \( \exists \) are different:

\[
\begin{align*}
t & \models \exists X. \psi \quad \text{iff there is a chain } C \text{ such that } \ t[X := C] \models \psi
\end{align*}
\]

- Obviously \( \text{FOL} \subseteq \text{CL} \subseteq \text{MSOL} \).
- A tree property definable in CL (but not in FOL) is: “there exists a path of even length”.
Definition (Df): A set of tree vertices $C$ is a *chain* iff it is totally ordered by the relation $\leq$.

*Chain logic* (CL) has the same syntax as monadic second order logic, but the semantics for the monadic quantifier $\exists$ are different:

$$ t \models \exists X.\psi \iff \text{there is a chain } C \text{ such that } t[X := C] \models \psi $$

- Obviously $\text{FOL} \subseteq \text{CL} \subseteq \text{MSOL}$.

- A tree property definable in CL (but not in FOL) is: “there exists a path of even length”.

- A regular tree property not definable in CL is: “the tree has an even number of vertices”.
Plan B

Our unattained plan B is two answer the question:

Given a regular tree language \( L \) decide whether \( L \) is chain definable.
Aperiodic tree languages

$t[]$: a tree with a hole.
Aperiodic tree languages

- $t[]$: a tree with a hole.
- $t[t']$: the substitution of some tree $t'$ into the hole.
Aperiodic tree languages

- $t[]$: a tree with a hole.
- $t[t']$: the substitution of some tree $t'$ into the hole
- Given a tree with a hole $t[]$, we define $t^1[] = t[]$, $t^n[] = t[t^{n-1}[]]$
Aperiodic tree languages

- $t[]$: a tree with a hole.
- $t[t']$: the substitution of some tree $t'$ into the hole

Given a tree with a hole $t[]$, we define $t^1[] = t[]$, $t^n[] = t[t^{n-1}[]]$

**Df:** A language is *aperiodic* if there is some $n \in \mathbb{N}$ such that for every tree with a hole $t[]$ and every tree $t'$, the trees $t^n[t']$ and $t^{n+1}[t']$ have the same type.
Aperiodic tree languages

- $t[]$: a tree with a hole.
- $t[t']$: the substitution of some tree $t'$ into the hole.
- Given a tree with a hole $t[]$, we define $t^1[] = t[]$, $t^n[] = t[t^{n-1}[]]$

**Df:** A language is *aperiodic* if there is some $n \in \mathbb{N}$ such that for every tree with a hole $t[]$ and every tree $t'$, the trees $t^n[t']$ and $t^{n+1}[t']$ have the same type.

**Fact:** [Potthoff 95] All FOL definable languages are aperiodic.
**Aperiodic tree languages**

- $t[]$: a tree with a hole.
- $t[t']$: the substitution of some tree $t'$ into the hole.
- Given a tree with a hole $t[]$, we define $t^1[] = t[]$, $t^n[] = t[t^{n-1}[]]$

**Df:** A language is *aperiodic* if there is some $n \in \mathbb{N}$ such that for every tree with a hole $t[]$ and every tree $t'$, the trees $t^n[t']$ and $t^{n+1}[t']$ have the same type.

**Fact:** [Potthoff 95] All FOL definable languages are aperiodic.

**Fact:** [Potthoff 95] Not all aperiodic languages are FOL definable.
Potthoff example (simplified)

One operator $\otimes$. Leaves labelled with 0, 1. All triples but the below two evaluate to $\bot$, which propagates.

\[
\begin{array}{cc}
\begin{array}{c}
1 \\
\otimes \\
0 \\
 0
\end{array} & \begin{array}{c}
0 \\
\otimes \\
1 \\
 1
\end{array}
\end{array}
\]
Potthoff example (simplified)

One operator $\otimes$. Leaves labelled with 0, 1. All triples but the below two evaluate to $\bot$, which propagates.

Let $L_T$ be the set of trees evaluating to $\tau \in \{0, 1, \bot\}$. 

\[
\begin{array}{c}
\begin{array}{c}
\otimes \\
0 \\
0
\end{array} & \quad & \\
\begin{array}{c}
\otimes \\
1 \\
1
\end{array}
\end{array}
\]
Potthoff example (simplified)

One operator $\otimes$. Leaves labelled with 0, 1. All triples but the below two evaluate to \( \perp \), which propagates.

\[
\begin{array}{c}
\frac{1}{\otimes} \\
0 & 0
\end{array}
\quad \quad
\begin{array}{c}
\frac{0}{\otimes} \\
1 & 1
\end{array}
\]

Let \( L_{\tau} \) be the set of trees evaluating to \( \tau \in \{0, 1, \perp\} \).

\( L_1 \cup L_{\perp} \) is the language of trees such that either: the leftmost path is of even length and ends in 0 or is of odd length and ends in 1.
Potthoff example (simplified)

One operator $\otimes$. Leaves labelled with 0, 1. All triples but the below two evaluate to $\bot$, which propagates.

$$\begin{align*}
\begin{array}{c}
1 \\
\otimes
\end{array} & \begin{array}{c}
0 \\
\otimes
\end{array} \\
0 & 0 & 1 & 1
\end{align*}$$

- Let $L_\tau$ be the set of trees evaluating to $\tau \in \{0, 1, \bot\}$.
- $L_1 \cup L_\bot$ is the language of trees such that either: the leftmost path is of even length and ends in 0 or is of odd length and ends in 1.
- $L_\bot$ is the language of trees such that some vertex within has one son in $L_1 \cup L_\bot$ and the other in $L_0 \cup L_\bot$. 
Potthoff example (simplified)

One operator $\otimes$. Leaves labelled with 0, 1. All triples but the below two evaluate to $\perp$, which propagates.

$$
\begin{align*}
\begin{array}{c}
\otimes \\
0 & \otimes \\
0 & 1 & 1
\end{array}
\end{align*}
$$

- Let $L_\tau$ be the set of trees evaluating to $\tau \in \{0, 1, \perp\}$.
- $L_1 \cup L_\perp$ is the language of trees such that either: the leftmost path is of even length and ends in 0 or is of odd length and ends in 1.
- $L_\perp$ is the language of trees such that some vertex within has one son in $L_1 \cup L_\perp$ and the other in $L_0 \cup L_\perp$.
- $L_0 = (L_0 \cup L_\perp) \setminus L_\perp$
Potthoff example (simplified)

One operator \( \otimes \). Leaves labelled with 0, 1. All triples but the below two evaluate to \( \bot \), which propagates.

\[
\begin{array}{c}
\begin{array}{c}
1 \\
\otimes
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
0 \\
\otimes
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
0 \\
1
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
1 \\
1
\end{array}
\end{array}
\]

\( L_{\tau} \) be the set of trees evaluating to \( \tau \in \{0, 1, \bot\} \).

\( L_1 \cup L_{\bot} \) is the language of trees such that either: the leftmost path is of even length and ends in 0 or is of odd length and ends in 1.

\( L_{\bot} \) is the language of trees such that some vertex within has one son in \( L_1 \cup L_{\bot} \) and the other in \( L_0 \cup L_{\bot} \).

\( L_0 = (L_0 \cup L_{\bot}) \setminus L_{\bot} \)

Fact: \( L_0 \) is in CL, not in FOL and is aperiodic.
Fact: $L_0$ is in CL, not in FOL and is aperiodic.
**Fact:** $L_0$ is in CL, not in FOL and is aperiodic.

The Potthoff example contradicts the following conjectures:
Potthoff example continued

**Fact:** $L_0$ is in CL, not in FOL and is aperiodic.

The Potthoff example contradicts the following conjectures:

- A language is FOL definable iff it is aperiodic.
Potthoff example continued

**Fact:** $L_0$ is in CL, not in FOL and is aperiodic.

The Potthoff example contradicts the following conjectures:

- A language is FOL definable iff it is aperiodic
- A chain definable language is FOL definable iff it is aperiodic
Confusion

Let $\mathcal{A} = \langle Q, q_0, \delta \rangle$ be a deterministic bottom-up automaton. Consider a tree $t$ with a designated subset of leaves $V$ and a function $\sigma : V \to Q$. $t[s] \in Q$ is defined as the state assumed by $\mathcal{A}$ in the root of $t$ starting from state $\sigma(v)$ in leaves $v \in V$ and from $q_0$ in the remaining vertices.
Let $\mathcal{A} = \langle Q, q_0, \delta \rangle$ be a deterministic bottom-up automaton. Consider a tree $t$ with a designated subset of leaves $V$ and a function $\sigma : V \rightarrow Q$. $t[s] \in Q$ is defined as the state assumed by $\mathcal{A}$ in the root of $t$ starting from state $\sigma(v)$ in leaves $v \in V$ and from $q_0$ in the remaining vertices.

**Df:** Let $R \subseteq Q$. We say $\mathcal{A}$ contains $R$-confusion if there is a tree $t$ with a designated set of leaves $V$ such that for every $v \in V$ and every $q, q' \in R$, there is some assignment $\sigma : V \rightarrow R$ such that $t[\sigma[v := q]] = q'$. 
Example of confusion
Example of confusion
Example of confusion

\[
\begin{array}{c}
\lor \\
\lor \\
0 & 1 & 1 & 1
\end{array}
\]
Example of confusion
Example of confusion

\[
\begin{array}{c}
\wedge \\
\top \\
\vee \\
\top \\
\top \\
\top \\
0 \\
\end{array}
\quad
\begin{array}{c}
\wedge \\
\top \\
\vee \\
\top \\
0 \\
\vee \\
\top \\
\end{array}
\]
Example of confusion
**Confusion conjecture**

**Df:** A language $L$ *contains confusion* if the minimal deterministic bottom-up automaton recognizing $L$ contains confusion. Otherwise $L$ is *non-confusing.*
**Confusion conjecture**

**Df:** A language $L$ contains confusion if the minimal deterministic bottom-up automaton recognizing $L$ contains confusion. Otherwise $L$ is non-confusing.

**Thm:** A chain definable language is non-confusing
Confusion conjecture

**Df:** A language $L$ contains confusion if the minimal deterministic bottom-up automaton recognizing $L$ contains confusion. Otherwise $L$ is non-confusing.

**Thm:** A chain definable language is non-confusing

**Conjecture:** A language is chain definable iff it is non-confusing
Arguments in favor of the conjecture

Works for languages with two types (i.e. whose minimal deterministic bottom-up automaton has two states)

Works for yield languages

Nonconflation behaves like a logic.
Arguments in favor of the conjecture

- Works for languages with two types (i.e. whose minimal deterministic bottom-up automaton has two states)
Arguments in favor of the conjecture

- Works for languages with two types (i.e., whose minimal deterministic bottom-up automaton has two states)
- Works for yield languages
Arguments in favor of the conjecture

- Works for languages with two types (i.e. whose minimal deterministic bottom-up automaton has two states)
- Works for yield languages
- Nonconfusion behaves like a logic.
Yield languages

Df: The *yield* $y(t)$ of a tree $t$ is the word consisting of the labels in the leaves of $t$, read from left to right.

Df: Let $L$ be a word language. A tree language of the form $\{t : y(t) \in L\}$ is called a *yield language*.
Yield languages

**Def:** The *yield* $y(t)$ of a tree $t$ is the word consisting of the labels in the leaves of $t$, read from left to right.

**Def:** Let $L$ be a word language. A tree language of the form \( \{ t : y(t) \in L \} \) is called a *yield language*.

**Thm:** A yield language is in CL iff it is in FOL iff it is non-confusing.
Nonconfusion behaves like a logic

**Thm:** Nonconfusing languages are closed under homomorphic images, direct and wreath products.
Nonconfusion behaves like a logic

**Thm:** Nonconfusing languages are closed under homomorphic images, direct and wreath products.

**Cor:** Nonconfusing languages are closed under boolean operations and chain quantification.
Simple algebras

1. Take the minimal deterministic bottom-up automaton \( \mathcal{A} \) recognizing \( L \). \( \mathcal{A} \) is non-confusing.
Simple algebras

1. Take the minimal deterministic bottom-up automaton $A$ recognizing $L$. $A$ is non-confusing.

2. Find a congruence $\sim$ in $A$. Then $A = A' \circ A_{/\sim}$ for some automaton $A'$. Both automata $A'$, $A_{/\sim}$ have fewer states. $L$ is non-confusing iff both $L(A')$ and $L(A_{/\sim})$ are chain definable.
Simple algebras

1. Take the minimal deterministic bottom-up automaton \( \mathcal{A} \) recognizing \( L \). \( \mathcal{A} \) is non-confusing.

2. Find a congruence \( \sim \) in \( \mathcal{A} \). Then \( \mathcal{A} = \mathcal{A}' \circ \mathcal{A}/\sim \) for some automaton \( \mathcal{A}' \). Both automata \( \mathcal{A}' \), \( \mathcal{A}/\sim \) have fewer states. \( L \) is non-confusing iff both \( L(\mathcal{A}') \) and \( L(\mathcal{A}/\sim) \) are chain definable.

3. Go back to 1.
Simple algebras

1. Take the minimal deterministic bottom-up automaton $A$ recognizing $L$. $A$ is non-confusing.

2. Find a congruence $\sim$ in $A$. Then $A = A' \circ A/\sim$ for some automaton $A'$. Both automata $A'$, $A/\sim$ have fewer states. $L$ is non-confusing iff both $L(A')$ and $L(A/\sim)$ are chain definable.

3. Go back to 1.

The base case: There is no congruence in $A$ ($A$ is a simple algebra).
Separation

The $L$-type of a tree $t$ is the state assumed in the root of $t$ by the minimal deterministic bottom-up automaton recognizing $L$.

An automaton $\mathcal{A}$ separates two types $\tau, \sigma$ if $\mathcal{A}$ accepts all trees of type $\tau$ and rejects all trees of type $\sigma$. 
Separation

The $L$-type of a tree $t$ is the state assumed in the root of $t$ by the minimal deterministic bottom-up automaton recognizing $L$.

An automaton $A$ separates two types $\tau, \sigma$ if $A$ accepts all trees of type $\tau$ and rejects all trees of type $\sigma$.

**Conjecture** If no deterministic top-bottom automaton can separate any two types then no chain logic formula can separate any two types.
The $L$-type of a tree $t$ is the state assumed in the root of $t$ by the minimal deterministic bottom-up automaton recognizing $L$.

An automaton $\mathcal{A}$ separates two types $\tau, \sigma$ if $\mathcal{A}$ accepts all trees of type $\tau$ and rejects all trees of type $\sigma$.

**Conjecture** If no deterministic top-bottom automaton can separate any two types then no chain logic formula can separate any two types.

**Fact:** If no deterministic top-bottom automaton can separate any two types then boolean combination of such automata can do it.
Summary and future work

- Try to characterise other logics such as CTL, MPL
Summary and future work

- Try to characterise other logics such as CTL, MPL
- Understand simple algebras
Summary and future work

- Try to characterise other logics such as CTL, MPL
- Understand simple algebras
- Understand word-sum automata (the order approach)
Summary and future work

- Try to characterise other logics such as CTL, MPL
- Understand simple algebras
- Understand word-sum automata (the order approach)
- Do something easier