Learning of Primitive Formal Systems Defining Labeled Ordered Tree Languages via Queries

Tomoyuki Uchida¹, Satoshi Matsumoto ², Takayoshi Shoudai ³, Yusuke Suzuki ¹, Tetsuhiro Miyahara ¹

¹ Hiroshima City University, Japan, ² Tokai University, Japan, ³ Kyushu International University, Japan
1. Introduction
2. Primitive Formal Systems Defining Labeled Ordered Tree Language
   a. Term Tree Pattern
   b. Primitive Formal Ordered Tree System (pFOTS) New! and pFOTS Languages New!
3. Contributions New!
4. Conclusions and Future Work
Learning of Formal Systems Defining Graph Languages

Learning string grammars
Given a language \( L = \{ab, aabb, aaabbb, \ldots\} \), learn a string grammar \( \Delta \) such that \( L = L(\Delta, \rho) \), where
\[
\Delta = \{p(ab) \leftarrow, p(axb) \leftarrow p(x)\}
\]

Elementary Formal System (EFS)
[first-order logic programing system, Smullyan 1961; Arikawa+1992; Miyano+2000; Sakamoto+2003]

Learning graph grammars
Given a graph language \( L = \{\ldots\} \), learn a graph grammar \( \Gamma \) such that \( L = L(\Gamma, \rho) \), where
\[
\Gamma = \{p(\ldots) \leftarrow, p(\ldots) \leftarrow p(\ldots)\}
\]

Formal Graph System (FGS)
[first-order logic programming system, Uchida+1995]
Learning of Formal Systems Defining Graph Languages

We consider the efficient learnabilities via queries of formal systems defining ordered tree languages by using background knowledge.

1. Machine Learning strategies for large graph data is to create a new rule representing replacements of subgraphs that are provable from background knowledge with appropriate variables.

2. Subgraph isomorphism problem for ordered trees is solvable in polynomial time.

3. Ordered trees can represent much graph data such as XML data, glycan data and parse structures of natural languages.

A logic program is suitable to represent!
Learning of Formal Systems Defining Graph Languages

We consider the efficient learnabilities via queries of formal systems defining ordered tree languages by using background knowledge.

\[ \text{Set of all ordered trees} \]

\[ \text{Target class} \]

\[ L(\Gamma \cup \{\alpha_2\}) \]

\[ L(\Gamma \cup \{\alpha_1\}) \]

\[ L(\Gamma \cup \{\alpha_2^*\}) \]

\[ \text{Query Learning} \]

Learner(Computer)

Oracle(Experts)

\[ \{\text{yes} \text{ if the target contains } T, \text{ no otherwise}\} \]

\[ \Gamma \text{ is a forma system defining an ordered tree language as background knowledge.} \]
A term tree pattern $t = (V_t, E_t)$ over $\langle \Sigma, \Lambda \cup \chi \rangle$ is a node- and edge-labeled ordered tree having variables, which are edges labeled with variable labels.

- $V_t$ : vertex set with $\psi_t : V_t \rightarrow \Sigma$
- $E_t \in V_t \times V_t$ : edge set with $\varphi_t : E_t \rightarrow \Lambda \cup \chi$
- $\Sigma = \{a, b, c, d, \ldots\}$ : finite set of vertex labels
- $\Lambda = \{a, b, c, \ldots\}$ : finite or infinite set of edge labels
- $\chi = \{x, y, z, \ldots\}$ : infinite set of variable labels

A term tree pattern over $\langle \Sigma, \Lambda \cup \chi \rangle$ is linear if all variables have mutually distinct variable labels.

A primitive term tree pattern is a term tree pattern over $\langle \Sigma, \chi \rangle$ consisting of two nodes and one variable.
A substitution is a finite set of bindings for variable labels.

For a term tree pattern $t$ and a substitution $\theta$, a new ordered tree $t\theta$ is obtained by replacing variables of $t$ with ordered trees of bindings in $\theta$.

$$\theta = \{x := [t_1, [v_1, v_2]], y := [t_2, [w_1, w_2]], z := [t_3, [s_1, s_2]]\}$$
Primitive Formal Ordered Tree Systems (pFOTS)

A **primitive Formal Ordered Tree System (pFOTS)** $\Gamma$ is a finite set of primitive graph rewriting rules (definite clauses) over $\langle \Pi, \Sigma, \Lambda \cup \chi \rangle$ of the form

$$p(t) \leftarrow q_1(f_1), \ldots, q_n(f_n) \quad (n \geq 1) \quad \text{or} \quad p(t') \leftarrow \quad (n = 0)$$

where $t$ is a **linear term tree pattern** over $\langle \Sigma, \Lambda \cup \chi \rangle$, $t'$ is an ordered tree over $\langle \Sigma, \Lambda \rangle$, $f_1, \ldots, f_n$ are **primitive term tree patterns** over $\langle \Sigma, \Lambda \cup \chi \rangle$ and $p, q_1, \ldots, q_n$ are unary predicate symbols in $\Pi$.

**pFOTS Language**

For a pFOTS $\Gamma$ over $\langle \Pi, \Sigma, \Lambda \cup \chi \rangle$ and a predicate symbol $p$ in $\Pi$, a **pFOTS Language** $L(\Gamma, p)$ is the set of all ordered trees $t$ such that the graph rewriting rule ‘$p(t) \leftarrow$’ is provable from $\Gamma$, that is,

$$L(\Gamma, p) = \{ \text{ordered tree } t \mid \Gamma \vdash p(t) \leftarrow \}.$$
Example of pFOTS and its pFOTS Language

\[
\Gamma_{OT} = \left\{ \begin{array}{l}
p(\alpha^o, x^2, c^o, y^2, b) \leftarrow p(\alpha^o, x^2, b), p(\alpha^o, y^2, b), \\
p(\alpha^o, x^2, b) \leftarrow p(\alpha^o, x^2, b), p(\alpha^o, y^2, b) \end{array} \right. \\
L(\Gamma_{OT}, p) = \\
\left\{ \begin{array}{l}
a \rightarrow b, a \rightarrow b, a \rightarrow c \rightarrow a \rightarrow b, a \rightarrow c \rightarrow b \rightarrow b, a \rightarrow b \rightarrow c \rightarrow c \rightarrow a \rightarrow b, \ldots \\
\ldots \\
\ldots \end{array} \right. 
\]
Target Classes of Query Learning Model

For a pFOTS \( \Gamma \) and a predicate symbol \( r \) that does not appear in \( \Gamma \), let \( \text{pGRR}(\Gamma, r) \) be the set of all primitive graph rewriting rules \( \alpha \) such that

- the predicate symbol in the head of \( \alpha \) is \( r \) and
- any predicate symbol in the body of \( \alpha \) appears in the head of a primitive graph rewriting rule in \( \Gamma \).

\[
\Gamma = \left\{ \begin{array}{c}
p(a \circ a \circ b) \leftarrow, \quad p(a \circ b \circ b) \leftarrow, \quad q(a \circ b \circ b) \leftarrow, \quad q(a \circ c \circ b) \leftarrow, \\
p(a \circ x \circ c \circ y \circ b) \leftarrow p(a \circ z \circ b), \quad p(a \circ y \circ b), \\
q(a \circ x \circ b) \leftarrow p(a \circ x \circ b), \quad q(a \circ y \circ b), \quad p(a \circ z \circ b)
\end{array} \right\}
\]

\( \text{pGRR}(\Gamma, r) \ni r(\quad ) \leftarrow q(a \circ x \circ b), \quad p(a \circ y \circ b), \quad q(a \circ z \circ b) \)
Target Classes of Query Learning Model

For a pFOTS $\Gamma$ and a predicate symbol $r$ that does not appear in $\Gamma$, let $\text{pGRR}(\Gamma, r)$ be the set of all primitive graph rewriting rules $\alpha$ such that

- the predicate symbol in the head of $\alpha$ is $r$ and
- any predicate symbol in the body of $\alpha$ appears in the head of a primitive graph rewriting rule in $\Gamma$.

We consider the learnabilities via queries of the classes

1. $\{L(\Gamma \cup \{\alpha\}, r) \mid \alpha \in \text{pGRR}(\Gamma, r)\}$

2. $\{L(\Gamma_1 \cup \cdots \cup \Gamma_K \cup \{\alpha\}, r) \mid \alpha \in \text{pGRR}(\Gamma_1 \cup \cdots \cup \Gamma_K, r)\}$

of sets of pFOTS languages for fixed pFOTSs $\Gamma, \Gamma_1, \cdots, \Gamma_K$, which are known in advance as background knowledge.
# Contributions and Related Work

| $|\Lambda| = 1$ | Polynomial-time inductive inference from positive data [Suzuki+2006(TCS)] | Open |
|----------------|--------------------------------------------------------------------------------|------|
| $2 \leq |\Lambda| \leq \infty$ | One Positive & Mem. Query [This work, Th. 1] | One Positive & Mem. Query If Cond. 2 holds [This work, Cor. 1] |
| $|\Lambda| = 1$ | One Positive & Mem. Query if Cond. 2 holds [This work, Cor. 2] | Identification in the limit by polynomial-time update from positive data & Mem. Query if Cond. 1 holds [Shoudai+2016(ILP)] |
| $K > 1, N = 1$ | Eq. Query & restricted Subset Query [Matsumoto+ 2008, Okada+2007] | |
| $K = 1, N > 1$ | | |

$\Pi(\Gamma)$ is the set of all predicate symbols in $\Gamma$

**BK:** $\Gamma_{OT}$

Target: rule $\alpha^*$

- $|\Pi(\Gamma)| = 1$
- $|\Pi(\Gamma)| \geq 2$

**BK:** $\Gamma^*$

Target: rules $\alpha_1^*, ..., \alpha_N^*$ where $|\Pi(\Gamma_k)| = 1$ ($1 \leq k \leq K$) and $\Pi(\Gamma_i) \cap \Pi(\Gamma_j) = \emptyset$ ($1 \leq i < j \leq K$)
### Contributions and Related Work

<table>
<thead>
<tr>
<th>$\Pi(\Gamma)$ is the set of all predicate symbols in $\Gamma$</th>
</tr>
</thead>
</table>

| $|\Lambda| = 1$ | BK: $\Gamma_{OT}$  
Target: rule $\alpha^*$ | BK: none  
Target: $\Gamma^*$ |
|---|---|---|
| 2 ≤ $|\Lambda| ≤ \infty$ | Polynomial-time inductive inference from positive data  
[Suzuki+2006(TCS)] | Identification in the limit by polynomial-time update from positive data & Mem. Query if Cond. 1 holds  
[Shoudai+2016(ILP)] |
Contributions and Related Work

** BK: \( \Gamma_{OT} \)**

**Target: rule \( \alpha^* \)**

Polynomial-time Inductive Inference from positive data
(Identification in the limit)

Identification in the limit by polynomial-time update from positive data & Mem. Query if Cond. 1 holds

\[ \text{Background Knowledge} \quad \Gamma_{OT} \]

\[ \text{defining the set of all ordered trees over } \{\{p\}, \{a, b, c\}, \{a, b\}\} \]

\[
\Gamma_{OT} \cup \{\alpha^*\} = \left\{ \begin{array}{l}
p(a^0 \text{ } a \text{ } b) \leftarrow, \quad p(a^0 \text{ } b \text{ } b) \leftarrow, \\
p(a^0 \text{ } x^2 \text{ } c^0 \text{ } y^2 \text{ } b) \leftarrow p(a^0 \text{ } x^2 \text{ } b), \quad p(a^0 \text{ } y^2 \text{ } b), \\
p(a^0 \text{ } x^2 \text{ } b) \leftarrow p(a^0 \text{ } x^2 \text{ } b), \quad p(a^0 \text{ } y^2 \text{ } b) \\
\end{array} \right. \]

Target rule \( \alpha^* \)
Contributions and Related Work

- Identification in the limit by polynomial-time update from positive data with membership query
- pFOTS in Chomsky normal form

Target pFOTS

\[ \Gamma^* = \{ \]

\[ p(a^o \ x^2 \ y^1 \ b) \leftarrow p(a^o \ x^2 \ b), q(a^o \ y^2 \ c), \]

\[ p(a^l \ x^2 \ b) \leftarrow p(a^l \ x^2 \ b), p(a^l \ y^2 \ b), \]

\[ q(a^l \ x^2 \ c) \leftarrow q(a^l \ x^2 \ c), q(a^l \ y^2 \ c) \]

\[ \} \]
### Contributions and Related Work

| BK: $\Gamma$, Target: rule $\alpha$ | BK: $\Gamma_1 \cup \ldots \cup \Gamma_K$ Target: rules where $|\Pi(\Gamma_k)| = 1$ ($1 \leq k \leq K$) $\Pi(\Gamma_i) \cap \Pi(\Gamma_j) = \emptyset$ ($1 \leq i < j \leq K$) |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $|\Pi(\Gamma)| = 1$ | $K > 1$, $N = 1$ |
| $|\Pi(\Gamma)| \geq 2$ | $K = \infty$ |
| $|A| = 1$ | Open |
| $2 \leq |A| \leq \infty$ | One Positive & Mem. Query [This work, Th. 1] |
| One Positive & Mem. Query If Cond. 2 holds [This work, Cor. 1] | One Positive & Mem. Query if Cond. 2 holds [This work, Cor. 2] |

*Eq. Query restricted*
Contributions and Related Work

\[ \begin{align*}
\text{BK: } \Gamma, \quad \text{Target: rule } \alpha^* \\
|\Pi(\Gamma)| = 1 & \quad \text{Query learning model} \\
|\Lambda| = 1 & \quad \text{• One Positive Example} \\
2 \leq |\Lambda| \leq \infty & \quad \text{• Membership Query} \\
\end{align*} \]

\[ \Gamma \cup \{\alpha^*\} = \begin{cases}
p(a \rightarrow b) \leftarrow, & p(a \rightarrow b) \leftarrow, \\
p(a \rightarrow x^2 \rightarrow c \rightarrow y^2 \rightarrow b) \leftarrow p(a \rightarrow x^2 \rightarrow b), p(a \rightarrow y^2 \rightarrow b), \\
p(a \rightarrow y^2 \rightarrow b) \leftarrow p(a \rightarrow x^2 \rightarrow b), p(a \rightarrow y^2 \rightarrow b), p(a \rightarrow z^2 \rightarrow b) \\
r(a \rightarrow b) \leftarrow p(a \rightarrow x^2 \rightarrow b), p(a \rightarrow y^2 \rightarrow b), p(a \rightarrow z^2 \rightarrow b)
\end{cases} \]

\[ \begin{align*}
&\text{One Positive \& Mem. Query} \\
&\text{[This work, Th. 1]} \\
&\text{One Positive \& Mem. Query} \\
&\text{[This work, Cor. 1]} \end{align*} \]

Background Knowledge \( \Gamma \)
Main Result
- Learnability of the class \( \{ L(\Gamma \cup \{ \alpha \}, r) \mid \alpha \in \text{pGRR}(\Gamma, r) \} \) via queries -

**Theorem 1.**

- Let \( \Gamma \) be a fixed pFOTS over \( \langle \Pi, \Sigma, \Lambda \cup \chi \rangle \) with \(|\Lambda| \geq 2\) and \(|\Pi(\Gamma)| = 1\),
  where \( \Pi(\Gamma) \) is the set of predicate symbols appearing in \( \Gamma \).
- Let \( r \) be a predicate symbol in \( \Pi \) that does not appear in \( \Gamma \), i.e., \( r \in \Pi \setminus \Pi(\Gamma) \).

Let \( \alpha^* \) be the target in \( \text{pGRR}(\Gamma, r) \). A primitive graph rewriting rule \( \alpha \in \text{pGRR}(\Gamma, r) \) such that

\[
L(\Gamma \cup \{ \alpha \}, r) = L(\Gamma \cup \{ \alpha^* \}, r)
\]

is exactly identified using \( O(N^2) \) membership queries and one positive example, where \( N \) is the number of edges in \( t \).
Contributions and Related Work

3(b). Contributions and Related Work

One Positive & Mem. Query
If Cond. 2 holds
[This work, Cor. 1]

BK satisfies Condition 2
(explained in the poster session)

Open

One Positive Example
• Membership Query

Background Knowledge Γ
Target: rule α*

BK: Γ, Target: rule α*
|Π(Γ)| ≥ 2

If Cond. 2 holds
[This work, Cor. 1]

One Positive & Mem. Query

Target: rule α∗
 Contributions and Related Work

- Query learning model
  - One Positive Example
  - Membership Query
- BK $\Gamma_{OT}$ and $\Gamma$ satisfy Condition 2

$$\Gamma_{OT} \cup \Gamma \cup \{\alpha^*\} =$$

**Background Knowledge $\Gamma_{OT}$**

- $p(a^o b b) \leftarrow, p(a^o b b) \leftarrow$
- $p(a^o x^2 c^o y^2 b) \leftarrow p(a^o x^2 b), p(a^o y^2 b)$
- $p(a^o x^2 b) \leftarrow p(a^o x^2 b), p(a^o y^2 b)$

**Background Knowledge $\Gamma$**

- $q(a^o c b) \leftarrow, q(a^o c b) \leftarrow$
- $q(a^o x^2 c^o y^2 b) \leftarrow q(a^o x^2 b), q(a^o y^2 b)$
- $q(a^o y^2 b) \leftarrow q(a^o x^2 b), q(a^o y^2 b), q(a^o z^2 b)$

**Target rule $\alpha^*$**

- $r(a^o z^2 b) \leftarrow q(a^o x^2 b), p(a^o y^2 b), q(a^o z^2 b)$

**Target**

- Rule $\alpha^*$

**Open**

- One Positive & Mem. Query if Cond. 2 holds

[This work, Cor. 2]

**BK**

- $\Gamma_1 \cup \cdots \cup \Gamma_K$, Target: rule $\alpha^*$

- $K > 1, N = 1$
## Conclusions and Future Work

| \(|\Lambda| = 1\) | \(2 \leq |\Lambda| \leq \infty\) |
|-------------------|-----------------|
| **BK: \(\Gamma_{OT}\)** |
| **Target: rule \(\alpha^*\)** |
| Polynomial-time inductive inference from positive data [Suzuki+2006(TCS)] | |
| **BK: \(\Gamma\)**, **Target: rule \(\alpha^*\)** |
| \(|\Pi(\Gamma)| = 1\) | \(|\Pi(\Gamma)| \geq 2\) |
| Open | | | | | | | |

### One Positive & Mem. Query if Cond. 2 holds [Cor. 1]

### Identification in the limit by polynomial-time update from positive data & Mem. Query if Cond. 1 holds [Shoudai+2016(ILP)]
### Conclusions and Future Work

| $|\Lambda| = 1$ | **BK:** $\Gamma_{OT}$  
Target: rule $\alpha^*$ | **BK:** $\Gamma$,  
Target: rule $\alpha^*$ |
|---|---|---|
| Polynomial-time inductive inference from positive data  
[Suzuki+2006(TCS)] | $|\Pi(\Gamma)| = 1$ | $|\Pi(\Gamma)| \geq 2$ |
| $2 \leq |\Lambda| \leq \infty$ | **Open** | **Open** |
| **One Positive & Mem. Query**  
[Th. 1] | **One Positive & Mem. Query**  
If Cond. 2 holds  
[Cor. 1] |

| $|\Lambda| = 1$ | **BK:** $\Gamma_1 \cup \cdots \cup \Gamma_K$  
Target: rules $\alpha_1^*, \ldots, \alpha_N^*$  
where $|\Pi(\Gamma_k)| = 1$ ($1 \leq k \leq K$) and  
$\Pi(\Gamma_i) \cap \Pi(\Gamma_j) = \emptyset$ ($1 \leq i < j \leq K$) | **BK:** none  
Target: $\Gamma$ |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$K &gt; 1, N = 1$</td>
<td>$K = 1, N &gt; 1$</td>
<td><strong>Open</strong></td>
</tr>
</tbody>
</table>
| **One Positive & Mem. Query if Cond. 2 holds**  
[Cor. 2] | **Identification in the limit**  
by polynomial-time update  
from positive data & Mem. Query if Cond. 1 holds  
[Shoudai+2016(ILP)] |

| $|\Lambda| = \infty$ | **Eq. Query & restricted Subset Query**  
[Matsumoto+ 2008, Okada+2007] |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Open</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4. Future Work
THANK YOU FOR YOUR ATTENTION!

If you’d like to know more information, please ask us in the poster session.
Appendix!!
Motivation!

1. the efficient learning for graph classes using a logic programming system to design efficient graph mining algorithms for graph data.

2. formalization of our previous work using formal graph system that directly manipulate graphs.

3. and so on
Why is in Query learning model?

Query learning is a model of a process of data mining using interaction between computer and experts.

So, one of applications of our results is to design an efficient graph mining algorithm for several tree structured data such as XML files, glycan data and so on.
Algorithm 1 LEARNING-pFOTS

Input: A tree $t \in \mathcal{OT}$.
Output: A rule $\alpha \in p_{\mathcal{GRR}}(\Gamma, r)$ such that $\mathcal{L}(\Gamma \cup \{\alpha\}, r) = \mathcal{L}(\Gamma \cup \{\alpha_\ast\}, q)$.
Assumption: A pFOTS $\Gamma$ is known as BK. Predicate symbols $r$ and $q$ are in $\Pi \setminus \Pi(\Gamma)$ and the target $\alpha_\ast$ is in $p_{\mathcal{GRR}}(\Gamma, q)$.

1: repeat
2: for all $f \in \{t' \in \mathcal{OT} \mid t \triangleright_\Gamma t'\}$ do
3: if MQ($f$) = yes then
4: $t := f$; break;
5: end if
6: end for
7: until $t$ does not change;
8: $f := t$; $n := 0$;
9: for all edges $e_f$ in $f$ do
10: Let $\mu$ be an edge label which is different from the edge label of $e_f$;
11: Let $f'$ be a tree obtained from $f$ by replacing $e_f$ with an edge labeled with $\mu$;
12: if MQ($f'$) = yes then
13: $n := n + 1$;
14: Replace the edge $e_t$ of $t$ that corresponds to $e_f$ with a new variable $h_n$;
15: end if
16: end for
17: // We regard the variable $h_i$ as a primitive term tree pattern consisting of $h_i$ ($i = 1, \ldots, n$).
18: output $\alpha := r(t) \leftarrow p(h_1), p(h_2), \ldots, p(h_n)$;
Algorithm 1: \textsc{LAERNING}\_pFOTS

\textbf{Input:} A tree $t \in \mathcal{OT}$

\textbf{Output:} A rule $\alpha \in p\mathcal{GRR}(\Gamma, r)$ with $\mathcal{L}(\Gamma \cup \{\alpha\}, r) = \mathcal{L}(\Gamma \cup \{\alpha^*_s\}, r)$

\textbf{Assumption:} A pFOTS $\Gamma$ is known. The target $\alpha^*_s$ belongs to $p\mathcal{GRR}(\Gamma, r)$.

\begin{enumerate}
  \item repeat
  \item for all $f \in \{t' \in \mathcal{OT} \mid t \triangleright_{\Gamma} t'\}$ do
  \item \hspace{1em} if $MQ(f) = \text{yes}$ then
  \item \hspace{2em} $t := f$; break;
  \item \hspace{1em} end if
  \item \hspace{1em} end for
  \item until $t$ does not change;
  \item $t := t$; $n := 0$;
  \item for all edges $e_f$ in $f$ do
  \item \hspace{1em} Let $\mu$ be an edge label which is different from the edge label of $e_f$;
  \item \hspace{1em} Let $f'$ be a tree obtained from $f$ by replacing $e_f$ with an edge labeled with $\mu$;
  \item \hspace{1em} if $MQ(f') = \text{yes}$ then
  \item \hspace{2em} $n := n + 1$;
  \item \hspace{2em} Replace the edge $e_t$ of $t$ that corresponds to $e_f$ with a new variable $h_n$;
  \item \hspace{1em} end if
  \item \hspace{1em} end for
  \item \hspace{1em} // Here $h_i$ denotes the primitive term tree pattern consisting of $h_i$ $(i = 1, \ldots, n)$.
  \item output $\alpha := r(t) \leftarrow p(h_1), p(h_2), \ldots, p(h_n)$;
\end{enumerate}
First Stage of Learning_pFOTS
Learning Algorithm

\[ \Gamma \cup \{ \alpha^* \} = \{ \]
\[ p(a \rightarrow b), p(a \rightarrow c), p(b \rightarrow c), p(c \rightarrow a), p(c \rightarrow b), p(a \rightarrow c), p(b \rightarrow c), p(c \rightarrow a), p(c \rightarrow b) \]
\[ \}

One Positive Example

\[ \in L(\Gamma \cup \{ \alpha^* \}, r) \]
Learning Algorithm

\[ \Gamma \cup \{ \alpha^* \} = \]

Matching

Background Knowledge \( \Gamma \)

Target rule \( \alpha^* \)
Learning Algorithm

\[ p(a \overset{a}{\to} b) \leftarrow p(a \overset{a}{\to} b), p(a \overset{a}{\to} b), p(a \overset{a}{\to} b), p(a \overset{a}{\to} b) \]

Background Knowledge \( \Gamma \)

\[ \Gamma \cup \{ \alpha^* \} = \]

Target rule \( \alpha^* \)

Matching
Learning Algorithm

\[ p(a_0 b) \leftarrow p(a_0 b), p(a_0 c), \]

Background Knowledge \( \Gamma \)

\[ \Gamma \cup \{\alpha^*\} = \]

Target rule \( \alpha^* \)

Matching

\[ r( \ldots ) \leftarrow p(a_0 x b), p(a_0 y b), p(a_0 z b) \]

Membership Query

\[ \in L(\Gamma \cup \{\alpha^*\}, r)? \]
Learning Algorithm

\[ \Gamma \cup \{ \alpha^* \} = \]

Matching

Background Knowledge \( \Gamma \)

Target rule \( \alpha^* \)

Membership Query

\[ \in L(\Gamma \cup \{ \alpha^* \}, r)? \]

Yes
\[
\Gamma \cup \{\alpha^*\} = \left\{ \begin{array}{l}
p(a \rightarrow b) \leftarrow, \quad p(a \rightarrow b) \leftarrow, \\
p(a \rightarrow c \rightarrow b) \leftarrow p(a \rightarrow b), \quad p(a \rightarrow c \rightarrow b), \\
p(a \rightarrow c \rightarrow b) \leftarrow p(a \rightarrow b), \quad p(a \rightarrow b), \quad p(a \rightarrow b) \\
r(\Gamma) \leftarrow p(a \rightarrow b), \quad p(a \rightarrow b), \quad p(a \rightarrow b) \end{array} \right. 
\]
Learning Algorithm

\[ \Gamma \cup \{\alpha^*\} = \{ \]

\[
\begin{align*}
p(a \rightarrow b) \leftarrow, & 
p(a \rightarrow b) \leftarrow, \\
p(a \rightarrow b) \leftarrow & \ p(a \rightarrow b), p(a \rightarrow b), \\
p(a \rightarrow b) \leftarrow & \ p(a \rightarrow b), p(a \rightarrow b), p(a \rightarrow b), \\
p(a \rightarrow b) \leftarrow & \ p(a \rightarrow b), p(a \rightarrow b), p(a \rightarrow b), \\
p(a \rightarrow b) \leftarrow & \ p(a \rightarrow b), p(a \rightarrow b), p(a \rightarrow b).
\end{align*}
\]

\[ \]
Learning Algorithm

\[
\Gamma \cup \{\alpha^*\} = \\
p(a^o \rightarrow b) \leftarrow, \ p(a^o \rightarrow b) \leftarrow, \\
p(a^o \rightarrow c \rightarrow b) \leftarrow p(a^o \rightarrow b), \ p(a^o \rightarrow b), \\
p(a^o \rightarrow c \rightarrow b) \leftarrow p(a^o \rightarrow b), \ p(a^o \rightarrow b), \ p(a^o \rightarrow b), \\
\]

Target rule \( \alpha^* \)
Learning Algorithm

\[ \Gamma \cup \{\alpha^*\} = \{\]
\[ p(\circ a \cdots b) \leftarrow, \quad p(\circ b \cdots b) \leftarrow, \]
\[ p(\circ a \cdots c \circ b) \leftarrow p(\circ a \cdots b), \quad p(\circ c \cdots b), \]
\[ p(\circ a \cdots c \circ b) \leftarrow p(\circ a \cdots b), \quad p(\circ c \cdots b), \quad p(\circ c \cdots b), \]
\[ r(\cdots) \leftarrow p(\circ a \cdots b), \quad p(\circ c \cdots b), \quad p(\circ c \cdots b), \quad p(\circ a \cdots b) \]
\[ \} \]

Background Knowledge \( \Gamma \)

Target rule \( \alpha^* \)
Learning Algorithm

Background Knowledge $\Gamma$

$\Gamma \cup \{\alpha^*\} = \{\}$

$$p(\text{a}, \text{a}) \leftarrow p(\text{a}, \text{b}) \leftarrow,$$
$$p(\text{a}, \text{c}, \text{b}) \leftarrow p(\text{a}, \text{b}), \ p(\text{a}, \text{c}, \text{b}),$$
$$p(\text{c}, \text{b}) \leftarrow p(\text{a}, \text{c}, \text{b}), \ p(\text{a}, \text{b}), \ p(\text{a}, \text{b}),$$
$$r(\text{a}, \text{b}, \text{b}, \text{c}) \leftarrow p(\text{a}, \text{c}, \text{b}), \ p(\text{a}, \text{b}), \ p(\text{a}, \text{b})$$

Hypothesis can not update no more in first stage!
Second Stage of Learning_pFOTS
Learning Algorithm

\[ \Gamma \cup \{ \alpha^* \} = \left\{ \begin{array}{l}
    p(a \rightarrow b) \leftarrow, \ p(a \rightarrow b) \leftarrow, \\
    p(a \rightarrow c) \leftarrow p(a \rightarrow b), \ p(a \rightarrow b), \ p(a \rightarrow c), \\
    p(a \rightarrow b) \leftarrow p(a \rightarrow b), \ p(a \rightarrow b), \ p(a \rightarrow c), \\
    r(a \rightarrow b) \leftarrow p(a \rightarrow b), \ p(a \rightarrow b), \ p(a \rightarrow b) 
\end{array} \right. \]

Background Knowledge \( \Gamma \)

Target rule \( \alpha^* \)

Hypothesis
Learning Algorithm

\[
p(a \xrightarrow{a} b) \leftarrow, \quad p(a \xrightarrow{b} c) \leftarrow, \\
p(a \xrightarrow{c} d) \leftarrow p(a \xrightarrow{b} c), \quad p(a \xrightarrow{d} e), \\
p(a \xrightarrow{e} f) \leftarrow p(a \xrightarrow{d} e), \quad p(a \xrightarrow{b} c), \quad p(a \xrightarrow{d} e)
\]

\[
\Gamma \cup \{\alpha^*\} = \{ \}
\]

Background Knowledge \(\Gamma\)

Target rule \(\alpha^*\)

\[
\alpha^* \subseteq L(\Gamma \cup \{\alpha^*\}, r) \Rightarrow \text{No}
\]

Hypothesis
\[ \Gamma \cup \{ \alpha^* \} = \begin{cases} p(\alpha^o \overset{a}{\rightarrow} b), p(\alpha^o \overset{b}{\rightarrow} b), & \text{Background Knowledge} \\ p(\alpha^o \overset{a}{\rightarrow} c \overset{c}{\rightarrow} b), p(\alpha^o \overset{b}{\rightarrow} b), p(\alpha^o \overset{c}{\rightarrow} b), & \text{Target rule } \alpha^* \end{cases} \]
Learning Algorithm

\[
\Gamma \cup \{\alpha^*\} = 
\begin{cases} 
p(\text{a} \rightarrow \text{b}), 
p(\text{a} \rightarrow \text{c} \rightarrow \text{b}), 
\end{cases}
\]

Background Knowledge \(\Gamma\)

Target rule \(\alpha^*\)

\[
\Gamma \cup \{\alpha^*\} = 
\begin{cases} 
p(\text{a} \rightarrow \text{b}), 
p(\text{a} \rightarrow \text{c} \rightarrow \text{b}), 
\end{cases}
\]

Hypothesis

\[
\in L(\Gamma \cup \{\alpha^*\}, r) \quad \text{?} 
\]

Yes
\[ \Gamma \cup \{ \alpha^* \} = \{ \]
\[ p(\alpha^o \rightarrow ^a \beta) \leftarrow, \quad p(\alpha^o \rightarrow ^b \beta) \leftarrow, \]
\[ p(\alpha^o \rightarrow ^c \beta) \leftarrow p(\alpha^o \rightarrow ^a \beta), \quad p(\alpha^o \rightarrow ^b \beta), \quad p(\alpha^o \rightarrow ^c \beta), \]
\[ p(\alpha^o \rightarrow ^c \beta) \leftarrow p(\alpha^o \rightarrow ^b \beta), \quad p(\alpha^o \rightarrow ^b \beta), \quad p(\alpha^o \rightarrow ^c \beta) \leq \]
\[ r(\quad \leftarrow \quad p(\alpha^o \rightarrow ^c \beta), \quad p(\alpha^o \rightarrow ^b \beta), \quad p(\alpha^o \rightarrow ^c \beta) \]
\[ \}

Learning Algorithm

Background Knowledge \( \Gamma \)

Target rule \( \alpha^* \)

Membership Query

\[ \epsilon \in L(\Gamma \cup \{ \alpha^* \}, r) ? \]

Yes
Learning Algorithm

\[ \Gamma \cup \{\alpha^*\} = \begin{cases} p(a \rightarrow b), & p(a \rightarrow c), \\ p(a \rightarrow b) \leftarrow p(a \rightarrow c), & p(a \rightarrow b), & p(a \rightarrow c), \end{cases} \]

Background Knowledge\(\Gamma\)

Target rule\(\alpha^*\)

Hypothesis
Learning Algorithm

\[ p(\overrightarrow{a 
 b}) \leftarrow, \quad p(\overrightarrow{a 
 b}) \leftarrow, \quad p(\overrightarrow{a 
 b}) \leftarrow p(\overrightarrow{a 
 b}), \quad p(\overrightarrow{a 
 b}), \quad p(\overrightarrow{a 
 b}), \quad p(\overrightarrow{a 
 b}), \quad p(\overrightarrow{a 
 b}) \]

\[ \Gamma \cup \{\alpha^*\} = \{ p(\overrightarrow{a 
 b}), \quad p(\overrightarrow{a 
 b}), \quad p(\overrightarrow{a 
 b}) \} \]

Background Knowledge \( \Gamma \)

Target rule \( \alpha^* \)

Membership Query

\( \in L(\Gamma \cup \{\alpha^*\}, r)? \)

Hypothesis

this process applies to all edges
\[ \Gamma \cup \{\alpha^*\} = \{ \]
\[ p(a\overrightarrow{a}b), p(a\overrightarrow{a}b), p(a\overrightarrow{a}b) \]
\[ p(a\overrightarrow{a}c\overrightarrow{a}b) \}
\[ r(\overrightarrow{a}b) \]