A Learning Algorithm for a Subclass of Tree Rewriting Systems

M. Jayasrirani\textsuperscript{1}, D.G. Thomas\textsuperscript{2}, Atulya K. Nagar\textsuperscript{3} and T. Robinson\textsuperscript{3}

\textsuperscript{1}Arignar Anna Government Arts College, Walajapet, India
\textsuperscript{2}Madras Christian College, Chennai - 600 059, India
\textsuperscript{3}Department of Computer Science, Liverpool Hope University, United Kingdom
Introduction

- Church-Rosser tree rewriting systems an alternative to describe and manipulate context-sensitive tree languages.

- Church-Rosser tree rewriting systems have many interesting properties such as word problem, language description of congruence classes, etc.

- An algorithm for learning a subclass of the class of Church-Rosser tree rewriting systems is given.

- Learning is obtained using membership queries.

- A teacher answers the membership queries related to the congruence classes made by the learner.
Definition

- $X = \{ x_1, x_2, \ldots \}$ a countable set of variables. $\Sigma$ is a ranked alphabet with symbols of different arities. $T_\Sigma(X)$ is the set of trees with variables from $X$.

- A set of rules $S$ over $\Sigma$ is a subset of $T_\Sigma(X) \times T_\Sigma(X)$. Each pair $(s, t)$ in $S$ is denoted as $s \rightarrow t$.

- The congruence generated by $S$ is the reflexive transitive closure $\Leftrightarrow^*_S$ of the relation $\Leftrightarrow_S$ defined as follows: For any two trees $t_1$ and $t_2$ in $T_\Sigma(X)$, if $t_2$ is obtained by matching a subtree $\bar{h}(s)$ of $t_1$ which is a substitution instance of one side of a rule in $S(s \rightarrow t, t \rightarrow s)$ and replacing it with the substitution instance of $\bar{h}(t)$ of the other side of that rule, then $t_1 \Leftrightarrow t_2$. 
Definition

- Given a set of rules $S$ over $\Sigma$, the relation $\Rightarrow_S$ is defined as $t \Rightarrow_S s$, if $t \Leftrightarrow s$ and $hg(t) > hg(s)$, $\forall t, s \in T_\Sigma(X)$.

- $\Rightarrow^*_S$ is the reflexive transitive closure of $\Rightarrow_S$.

- $(S, \Rightarrow_S)$ is called a tree replacement (rewriting) system on $\Sigma$.

- A tree $t$ is irreducible ($\text{mod } S$) if there is no tree $t'$ such that $t \Rightarrow_S t'$.

- $\text{IRR}(S)$ is the set of all irreducible trees with respect to $S$. 
  
  

A tree replacement system \((S, \Rightarrow_S)\) is Church-Rosser if for all trees \(t_1, t_2\) with \(t_1 \Leftrightarrow^*_S t_2\), there exists a tree \(t_3\) such that \(t_1 \Rightarrow^*_S t_3\) and \(t_2 \Rightarrow^*_S t_3\).

- The word problem for a tree replacement system \((S, \Rightarrow_S)\) is that given any two trees \(s, t\) in \(T_{\Sigma}(X)\), deciding whether \(s\) and \(t\) are congruent to each other or not.
- The word problem is undecidable in general for any tree replacement system.
- The word problem for any Church-Rosser tree replacement system is decidable.
Example

Let the trees $q, s, t, s', t', t_1$ and $t_2$ be in $T_\Sigma(X)$ where $\Sigma = \{a, b, c, d, x, y\}$ and $X = \{x, y\}$. Let $(s, t)$ or $(t, s)$ be a rule in $S$. Let $q = a(b(a(c, d), c), a(b(d, c), d))$ be a tree in $T_\Sigma(X)$ as shown in Figure.

Figure: Tree $q$
Example (contd...)

Let \( s = a(x, y) \) and \( t = a(c, b(y, x)) \) be two trees as shown in Figure.

![Tree s and t](image)

**Figure:** Trees \( s \) and \( t \)

![Tree s' and t'](image)

**Figure:** Trees \( s' \) and \( t' \)
Example (contd...)

Figure: Trees $t_1$ and $t_2$
Definition

A tree rewriting system $T$ on $\Sigma$ is called reduced if for every rewriting rule $(s, t) \in T$, $t$ is an irreducible tree with respect to $T$ and $s$ is an irreducible tree with respect to $T - \{(s, t)\}$.

Let $T$ be a tree rewriting system on $\Sigma$. For a tree $t \in T_\Sigma(X)$, $s$ is called a normal form of $t$, if $s \in [t]_T$ and $s$ is an irreducible tree with respect to $T$. 
Procedure for Learning Church-Rosser Tree Rewriting System $R$

- $T$ - Church-Rosser tree rewriting system on $\Sigma$.
- $M_T = \{L_1, L_2, \ldots, L_n\}$ - quotient monoid where each $L_i$ is a congruence class of a tree with respect to $T$.
- The congruence relation $\Leftrightarrow^*_T$ is of finite index and so each congruence class $L_i$ ($1 \leq i \leq n$) is a regular tree language.
- $M_T$ finite yields an efficient learning procedure for congruence classes with only membership queries.
- The unique reduced Church-Rosser tree rewriting system $R$ equivalent to $T$ is then obtained.
- The learning procedure to obtain $R$ consists of two parts, one for $\text{IRR}(R)$ and the other for the tree rewriting system $R$. 
For any tree $t \in T_\Sigma$ given as input, the oracle answers membership query by producing an $n$-tuple that contains $(n - 1)$ zeros and one 1 since $M_T = M_R = \{L_1, L_2, \ldots, L_n\}$.

The learner gets the value of $n$ when the empty tree $\Lambda$ is given as input for membership query.

The input is a tree $t \in T_\Sigma$ and the output is an $n$-tuple $q(t) = (k_1, k_2, \ldots, k_n)$ where $k_i = 1$ if $t \in L_i$ and $k_i = 0$ if $t \notin L_i$ ($1 \leq i \leq n$).
Irreducible trees in $T^0_{\Sigma}$

- Membership queries are made to the oracle for the input trees, starting with the empty tree $\Lambda$, which is an irreducible tree with respect to $R$ and continued with the trees in $T^0_{\Sigma}$.

- Let $t_1 = \Lambda$ and suppose $t_2, t_3, \ldots, t_s$ are the lexicographically ordered trees in $T^0_{\Sigma}$ where $(s - 1)$ is the number of constants in $\Sigma$.

- A tree $t_i$ (2 ≤ $i$ ≤ $s$) belonging to $L_j$ for some $j$ (1 ≤ $j$ ≤ $n$) is an irreducible tree with respect to $R$ whenever $t_i \in L_j$ but $t_p \notin L_j$ for $p = 1, 2, \ldots, i - 1$.

- Hence by membership queries all the irreducible trees in $T^0_{\Sigma}$ with respect to $R$ are obtained.
Other irreducible trees

- The process is continued by making membership queries for trees in $T^1_\Sigma (T^0_\Sigma \cap IRR(R))$, the set of all trees of height one with subtrees in $T^0_\Sigma \cap IRR(R)$, which can be lexicographically ordered.

- Thus the process gives irreducible trees with respect to $R$ in $T^0_\Sigma$ and $T^1_\Sigma$. In general the process is continued recursively by making membership queries for trees in $T^1_\Sigma (T^{r-1}_\Sigma \cap IRR(R))$, the set of all trees of height $r$, with subtrees in $T^{r-1}_\Sigma \cap IRR(R)$, $r \geq 1$. This process terminates when each $L_j$ receives an irreducible tree with respect to $R$.

- The algorithm for forming irreducible trees with respect to $R$, terminates when the process for finding trees with respect to $R$ in $T^k_\Sigma$ ends, where $k = \max \{hg(t) | t \in IRR(R)\}$ since (a) $IRR(R)$ is finite and (b) each $L_j$ ($1 \leq j \leq n$) contains exactly one irreducible tree with respect to $R$ and (c) irreducible trees with respect to $R$ are shortest trees in their respective classes $L_1, L_2, \ldots, L_n$. 
To identify the unique, reduced Church-Rosser tree rewriting system $R$ equivalent to the unknown tree rewriting system $T$, the learner performs again the membership queries as in the procedure for the lexicographically ordered trees in the set $T_1^1(IRR(R)) - IRR(R)$, where $T_1^1(IRR(R))$ is the set of all trees with subtrees in $IRR(R)$ in the next level.

The learner then forms the tree rewriting system

$$S = \left\{(s, t) \mid s \in T_1^1(IRR(T)) - IRR(T), t \in IRR(T), s \text{ and } t \text{ both belong to } L_j \text{ for some } j(1 \leq j \leq n)\right\} \text{ on } \Sigma$$

From $S$, a reduced tree rewriting system $S'$ equivalent to $S$ on $\Sigma$ is obtained and thus the learner obtains $R$ which is same as $S'$ on $\Sigma$. 

Learning $R$
Example

- $R = \{(b(c), c), (b(d), d), (a(c, c), c), (a(d, d), d),
  (a(c, d), c), (a(d, c), d)\}$ on $\Sigma = \{a, b, c, d\}$ with arities of $a, b, c, d$ as 2, 1, 0, 0 respectively.
- $M_R = \{[\Lambda]_R, [c]_R, [d]_R\}$.
- Membership queries are made for the trees $\Lambda, c, d$ belonging to $T^0_\Sigma$ and the oracle produces the answers $q(\Lambda) = (1, 0, 0), q(c) = (0, 1, 0), q(d) = (0, 0, 1)$ for which the learner obtains $IRR(R)$ as $\{\Lambda, c, d\}$.
- Again membership queries are made for the trees in the set $T^1_\Sigma = \{b(c), b(d), a(c, c), a(d, d), a(c, d), a(d, c)\}$ and the oracle produces the answers $q(b(c)) = (0, 1, 0), q(b(d)) = (0, 0, 1)$
  $q(a(c, c)) = (0, 1, 0), q(a(d, d)) = (0, 0, 1)$
  $q(a(c, d)) = (0, 0, 1), q(a(d, c)) = (0, 0, 1)$
- The learner obtains $S = \{(b(c), c), (b(d), d), (a(c, c), c), (a(d, d), d),
  (a(c, d), c), (a(d, c), d)\}$. \(15/24\)
Example

- $R = \{(b(c), c), (a(c, c), c)\}$ on $\Sigma = \{a, b, c\}$ with arities of $a, b, c$ as $2, 1, 0$ respectively.

- $M_R = \{[\Lambda]_R, [c]_R\}$.

- The trees in $T^0_\Sigma = \{\Lambda, c\}$.
  - $q(\Lambda) = (1, 0), q(c) = (0, 1)$

- The trees in $T^1_\Sigma = \{a(c, c), b(c)\}$.
  - $q(b(c)) = (0, 1), q(a(c, c)) = (0, 1)$

- The learner obtains
  - $S = \{(b(c), c), (a(c, c), c)\}$. 
Example

- \( R = \{(a(d, d, d), d), (b(d, d), d), (c(d), d)\} \) on \( \Sigma = \{a, b, c, d\} \) with arities of \( a, b, c, d \) as 3, 2, 1, 0 respectively.

- \( M_R = \{[\Lambda], [d]_R\} \).

- The trees in \( T^0_\Sigma = \{\Lambda, d\} \).
  \( q(\Lambda) = (1, 0), \ q(d) = (0, 1) \)

- The trees in \( T^1_\Sigma = \{a(d, d, d), b(d, d), c(d)\} \).
  \( q(b(d, d)) = (0, 1), \ q(a(d, d, d)) = (0, 1), \ q(c(d)) = (0, 1) \)

- The learner obtains
  \( S = \{(a(d, d, d), d), (b(d, d), d), (c(d), d)\} \).
Algorithm for Learning $IRR(R)$

begin

$IRR(R) = \emptyset$
Input the empty tree $t_1 = \Lambda$
$n = \text{number of entries in } q(\Lambda)$
$L_1 = \{\Lambda\}$
$IRR(R) = \{\Lambda\}$
$N_1 = 1$

For $j = 2$ to $n$, initialize: $L_j = \emptyset$; $N_j = 0$
Input trees $t_i (i = 2, 3, \ldots)$ ordered according to height (trees of same height are lexicographically ordered) such that
$t_i \in T^0_\Sigma \cup T^1_\Sigma (T^{r-1}_\Sigma \cap IRR(R)), (r \geq 1)$

while $N_j = 0$ for some $j$ do

begin

For $j = 1$ to $n$ do
begin
If $p_j(q(t_i)) = 1$ do
begin
$L_j = L_j \cup \{t_i\}$
If $N_j = 0$ do
begin
$N_j = 1$
$IRR(R) = IRR(R) \cup \{t_i\}$
end
end
end

output $IRR(R)$

end.
Algorithm for Learning $R$

begin

Input trees $t_i$ ($i = 1, 2, 3, \ldots$) ordered according to height (trees of same height are lexicographically ordered) such that

$t_i \in T^1_\Sigma(IRR(T)) - IRR(T)$

Initialize: $S = \emptyset$

For $s \in T^1_\Sigma(IRR(T)) - IRR(T)$ do

begin

For $t \in IRR(R)$ do

begin

If $p_j(q(s)) = p_j(q(t)) = 1$ for some $j$ ($1 \leq j \leq n$), then

$S = S \cup \{(s, t)\}$

end

end

begin

Initialize: $S' = S$

For $(s, t) \in S'$, do

begin

If $(s_1, t_1) \in S' - \{(s, t)\}$ such that $s$ has $s_1$ as a subtree, then

$S' = S' - \{(s, t)\}$

end

end

output: $R = S'$

end.
Lemma

\[ IRR(S) = IRR(R). \]

Lemma

\[ t_1 \leftrightarrow_S t_2 \implies t_1 \leftrightarrow_R t_2 \text{ for } t_1, t_2 \in T_\Sigma(X). \]

Lemma

**Cardinality of** \( M_S \) **is** \( n \). **That is** \( \text{card}(M_S) = n \).

Lemma

*For* \( s, t \in T_\Sigma(X) \), \( s \leftrightarrow_R t \) **implies** \( s \leftrightarrow_S t \).

Theorem

\( R \) **and** \( S \) **are equivalent.**
Lemma

For any tree rewriting systems $T$ that is Church-Rosser, there is a unique reduced tree rewriting system $T'$ that is Church-Rosser and equivalent to $T$. Furthermore, one can effectively construct $T'$ from $T$.

Lemma

Let $T$ and $T'$ be two equivalent tree rewriting systems. If $T$ is Church-Rosser and $\text{IRR}(T) = \text{IRR}(T')$, then $T'$ is also Church-Rosser.

Theorem

$S$ is Church-Rosser.

Theorem

$S' = R$ where $S'$ is a reduced tree rewriting system equivalent to $S$. 
The number of trees to be processed through membership query for learning $IRR(R)$ can be found.

The number of trees to be processed through membership query for learning $IRR(R)$ is less than or equal to $1 + m(n + 2)$ where $m = card(T_{\Sigma}^1)$ which is fixed and $n =$ total number of congruence classes with respect to $R$.

The trees to be processed are in the set
$$F = \{\Lambda\} \cup T_{\Sigma}^0 \cup T_{\Sigma}^1(T_{\Sigma}^0 \cap IRR(R)) \cup T_{\Sigma}^1(T_{\Sigma}^1 \cap IRR(R)) \cup T_{\Sigma}^1(T_{\Sigma}^2 \cap IRR(R)) \cup \cdots \cup T_{\Sigma}^1(T_{\Sigma}^{r-1} \cap IRR(R))$$
where $r = \max\{hg(t) | t \in IRR(R)\}$.

$card \ F \leq 1 + m(n + 2)$. 


Thank You