Characteristic Sets for Inferring the Unions of the Tree Pattern Languages by the Most Fitting Hypotheses

Yen Kaow Ng[1] and Takeshi Shinohara[2]

[1] Kyushu Institute of Technology
Graduate School of Computer Science and Systems, Iizuka, 820, Japan
kalngyk@daisy.ai.kyutech.ac.jp
[2] Kyushu Institute of Technology
Department of Artificial Intelligence, Iizuka, 820, Japan
shino@ai.kyutech.ac.jp

Abstract. A tree pattern p is a first-order term in formal logic, and the language of p is the set of all the tree patterns obtainable by replacing each variable in p with a tree pattern containing no variables. We consider the inductive inference of the unions of these languages from positive examples using strategies that guarantee some forms of minimality during the learning process. By a result in our earlier work, the existence of a characteristic set for each language in a class $\mathcal{L}$ (within $\mathcal{L}$) implies that $\mathcal{L}$ can be identified in the limit by a learner that simply conjectures a hypothesis containing the examples, that is minimal in the number of elements of up to an appropriate size. Furthermore, if there is a size ℓ such that each candidate hypothesis has a characteristic set (within the languages in $\mathcal{L}$ that intersects non-emptily with the examples) that consists only of elements of up to size ℓ, then the hypotheses containing the least number of elements of up to size ℓ are at the same time minimal with respect to inclusion. In this paper we show how to determine such a size ℓ for the unions of the tree pattern languages, and hence allowing us to learn the class using hypotheses that fulfill the two mentioned notions of minimality.

1 Introduction

In this paper we consider the *characteristic sets* [2] for the unions of the languages generated by the *(ordered, ranked and finite) tree patterns* [15]. A tree pattern is a first-order term in formal logic. Intuitively a tree pattern can be understood as a tree structure with leaves labeled with constant symbols or variables. The variables may be replaced with other tree patterns. A tree pattern is said to be *ground* if it contains no variable. The language of a tree pattern p are all the ground tree patterns obtainable by substituting variables in p with ground tree patterns. A *characteristic set S of a language L within a class $\mathcal{L}$* is a finite set of elements in L such that if any language in $\mathcal{L}$ includes S, then it must also include L.

Let $\mathcal{L}^k$ denote the class of unions of up to k tree pattern languages. We are interested in the characteristic sets of $\mathcal{L}^k$, for within the two classes $\mathcal{L}^k$ and $\mathcal{L}^k_T$. We now describe $\mathcal{L}^k_T$. For any given set T of ground tree patterns let $\mathcal{L}_T$ denote the class of tree pattern languages that intersect non-emptily with T. $\mathcal{L}^k_T$ is then, the class of unions of up to k languages from $\mathcal{L}_T$. We show in this paper how to compute, given a language $L \in \mathcal{L}^k$ and a class $\mathcal{L}'$ chosen from one of the following:

1. $\mathcal{L}^k$ where k is less than the alphabet size, and
2. $\mathcal{L}^k_T$ where k is independent of the alphabet size

a length ℓ_T, such that there is a characteristic set for L, consisting only of trees of up to the size ℓ_T, within $\mathcal{L}'$. The choice of $\mathcal{L}^k_T$ is due to an earlier result, which shall become clear later.

The present study is motivated by an interest to inductively infer, from positive examples [9], the unions of the tree pattern languages, using hypotheses that are minimal in some sense. In using such hypotheses, we are attempting to overcome the problem of over-generalization due to lack of negative examples. The learning of the class of the tree pattern languages is interesting since the tree patterns can also be thought of as abstractions of hierarchically structured data such as XML documents [10], which are becoming increasingly popular for use on the internet as well as for biological applications. Since the tree patterns are first-order terms in logic programs, this study can also be considered a first step towards the inductive inference of logic programs (e.g. Prolog programs) from positive data.

It has been previously shown that the unions of the tree pattern languages can be inferred efficiently when the number of languages allowed in a union is less than the alphabet size [4, 7, 6]. In this work we do not address the issue of efficiency in the learning process — our primary interest is to obtain hypotheses that possess some forms of minimality. In a previous work [13], we found that any class of languages $\mathcal{L} = \{L_i \mid i \in N^+\}$ where each language in the class has a characteristic set within the class, can be identified in the limit by the following learner `infer`, which basically outputs a hypothesis that contains the fewest elements of sizes up to the length of the input sequence. In the following, the learner is given as input the sequence σ.

`infer`(σ):
  If $|\sigma| = 0$, let $H = \emptyset$. (Initial hypothesis)
  If H is not among the languages containing all examples in σ with the fewest elements of sizes up to $|\sigma|$, find the language from among these minimal languages with the smallest index and set H to it.
    Note that the smallest index is used here to simplify convergence.
  Conjecture H.

The use of such minimal hypotheses is beneficial for the following reasons: (1) this reduces learning into a problem of counting the elements of the candidate hypotheses of up to some length (which is often computable), and (2) some works have suggested that such hypotheses may be desirable in some applications [8, 12, 14]. Note that `infer` is computable for the class of unions of tree pattern languages.

The previous work also showed that, if there exists a size ℓ such that, for every language L that is consistent with the input, there is a characteristic set of L (within $\mathcal{L}_T^k$) consisting only of elements of up to the size ℓ, then all the hypotheses containing the input that has the least number of elements of up to the size ℓ are also minimal with respect to inclusion (we restate this in Theorem 2). That is, we can obtain hypotheses that possess both of these notions of minimality. Note that hypotheses that are minimal with respect to inclusion possess a few interesting qualities [1], and may also help in the analysis of convergence of the learning process (via the characteristic sets).

This paper is structured as follows. In Section 2 we formally define the terminology and concepts needed for subsequent discussions. We define the two notions of minimality which we are interested in, and give earlier results on these notions of minimality. The motivation for our choice of classes that the characteristic sets are with respect to will become clear from this result. Section 3 discusses the existence of these characteristic sets. In Section 4 we show a sufficient size bound to the characteristic sets of the unions of the tree pattern languages, when the number of languages allowed in a union is less than the alphabet size. This restriction on the bound of the number of unions is the same as that in [4]. Finally, in Section 5, we study the characteristic sets with this restriction on the alphabet size removed.

2 Preliminaries

We follow mainly the definitions in [3]. The symbols N and N^+ denote the set of natural numbers and the set of positive integers respectively. $\emptyset$, $\in$, $\subset$, $\subseteq$ respectively denote empty set, element of, proper subset, subset. max(.), min(.) denote maximum and minimum of a set, where by convention $\max(\emptyset) = 0$ and $\min(\emptyset) = \infty$. Cardinality of a set S is written $card(S)$. $A - B$ denotes the set $\{x \mid x \in A \text{ and } x \notin B\}$. The symbol Σ denotes a finite alphabet, where each element of Σ is called a *function* and associated with a non-negative integer called an *arity*. A function with arity 0 is also called a *constant*. We assume that

Σ contains at least one constant. Let $V = \{x_1, x_2, \ldots\}$ be a countable set of symbols disjoint from Σ. Each element in V is called a variable.

2.1 Tree patterns

A *tree pattern* p over Σ is an expression defined recursively as follows: (1) a constant $f \in \Sigma$, or a variable $x \in V$ is a tree pattern; (2) For a function f with arity n, where $n \geq 1$, and tree patterns $t_1, \ldots, t_n$, $f(t_1, \ldots, t_n)$ is a tree pattern. The function f or variable x in (1) and (2) is the *root symbol* of p. We consider tree patterns to be defined from left to right, that is, in $f(t_1, \ldots, t_n)$ of (2), t_i is defined before t_j if $i < j$. A tree pattern t *occurs* in a tree pattern p just in case t is used in the recursive definition of p. We call each such occurrence of t a *node of p* (or simply *a node*). We say that a node u is the i-th *child* (or simply *a child*) of a node v of the term $f(t_1, \ldots, t_n)$, just in case u occurs as t_i $(1 \leq i \leq n)$ in v.

A tree pattern is said to be *ground* if it contains no variable. Let TP denote the set of all tree patterns and let TP_Σ denote the set of all ground tree patterns. The *size of a tree pattern* p, denoted by $\|p\|$, is the number of nodes in p (Note: this definition differs from that in [3]).

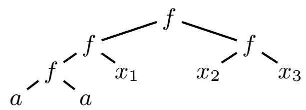
Given a tree pattern p and a node v in p, the *position of* v, written pos(v), is a string of natural numbers recursively defined as follows:

$$\mathrm{pos}(v) = \begin{cases} \epsilon \text{ (empty string)} & \text{if the node } v \text{ is } p \text{ itself.} \\ \mathrm{pos}(v')i & \text{if } v \text{ is the } i\text{-th child of the node } v' \text{ in } p. \\ \text{undefined} & \text{otherwise.} \end{cases}$$

We say that a function symbol c *occurs at a position* ε in p just in case pos$(v) = \varepsilon$ for some node v in p and c is the root symbol of v. Likewise a tree pattern t *occurs at a position* ε in p just in case pos$(v) = \varepsilon$ and v is an occurrence of t. The *height* of a tree pattern t, $|t|$, is $\max(\{|\mathrm{pos}(v)| \mid v \text{ is a node in } t\})$, where $|\mathrm{pos}(v)|$ is the length of the string pos(v). We say that a node v is at depth h just in case $|\mathrm{pos}(v)| = h$.

Finally, we let $\langle n \rangle_\Sigma$ denote the smallest size s such that there are n elements of the same height in $\{t \in TP_\Sigma \mid \|t\| \leq s\}$, if such elements exist.

Example 1 *Let $\Sigma = \{a, f\}$ where a is a constant and f is of arity 2. Consider $\langle 7 \rangle_\Sigma$. Let $p = f(f(f(a, a), x_1), f(x_2, x_3))$ (see below).*



By replacing either a or $f(a, a)$ in each of the variables in p (hence the maximum size of these tree patterns is 15), we can verify that there are at least 7 tree patterns (8 for this case) of equal height (that is, 3) in TP_Σ. Hence $\langle 7 \rangle_\Sigma \leq$ 15.

The following Proposition states that such n ground tree patterns of equal height exist under very trivial conditions.

Proposition 1 *If Σ consists of at least one function of arity ≥ 2, then for all n, there exists n elements of the same height in TP_Σ.*

2.2 Languages of the tree patterns

A *substitution* is a finite set of the form $\{y_1/t_1, \ldots, y_n/t_n\}$, where each y_i is a distinct variable in V, and each t_i is a tree pattern different from y_i. An *instance* of a tree pattern p by a substitution $\theta = \{y_1/t_1, \ldots, y_n/t_n\}$, written $p\theta$, is the tree pattern obtained by simultaneously replacing each occurrence of the variable y_i with the term t_i ($1 \leq i \leq n$). We write $p \preceq q$ iff $p = q\theta$ for some substitution θ. We write $p \prec q$ just in case $p \preceq q$ but not $q \preceq p$. Given two sets of tree patterns P and Q, we write $P \sqsubseteq Q$ iff for each $p \in P$, $p \preceq q$ for some $q \in Q$. $P \sqsubset Q$ just in case $P \sqsubseteq Q$ but not $Q \sqsubseteq P$.

The language of a tree pattern p, written $L(p)$, is the set $L(p) = \{w \in TP_\Sigma \mid w \preceq p\}$. For a set $P \subseteq TP$, $L(P) = \bigcup_{p \in P} L(p)$ and $\mathcal{L}(P) = \{L(p) \mid p \in P\}$. For any class $\mathcal{L}$ of languages, $\mathcal{L}^k$ denotes the class $\{\bigcup_{L \in \mathcal{L}'} L \mid \mathcal{L}' \subseteq \mathcal{L} \wedge card(\mathcal{L}') \leq k\}$, while $\mathcal{L}^*$ denotes the class $\{\bigcup_{L \in \mathcal{L}'} L \mid \mathcal{L}' \subseteq \mathcal{L} \wedge card(\mathcal{L}') < \infty\}$.

Definition 1 (Characteristic sets [2]) *A characteristic set for a language L within a class $\mathcal{L}$ is a finite set S_L such that for each $L' \in \mathcal{L}$, $S_L \subseteq L' \Rightarrow L \subseteq L'$.*

Definition 2 (S_n) *For any $p \in TP$, we denote by $S_n(p)$ the set of all the elements of $L(p)$ obtainable by replacing variables in p with ground tree patterns of* sizes up to n.

2.3 The MINL and FC notions of minimality

We now describe two problems of finding, given a finite non-empty set $S \subseteq TP_\Sigma$, a language $L \in \mathcal{L}(TP)^k$ containing S that fulfills some notions of minimality. Earlier studies [1, 17, 5, 13] have shown that solutions from either one these problems allow us to inductively infer any class of languages where each language in the class has a characteristic set within the class.

Definition 3 *Given a finite non-empty set $S \subset TP_\Sigma$ and $k \in N^+$,*
*(1) The problem **MINL**$(\mathcal{L}(TP), S, k)$ is to find a set of at most k tree patterns $P \subseteq TP$ such that $S \subseteq L(P)$ and there are no other set of up to k patterns $P' \subseteq TP$ where $S \subseteq L(P')$ such that $L(P') \subset L(P)$.*
*(2) The problem **FC**$(\mathcal{L}(TP), S, k, \ell)$ is to find, among every set P of up to k tree patterns in $\{p \in TP \mid L(p) \cap S \neq \emptyset\}$ where $S \subseteq L(P)$, for one that minimizes $card(\bigcup_{p \in P}\{t \in L(p) \mid \|t\| \leq \ell\})$.*
*(3) $\mathcal{MINL}(\mathcal{L}(TP), S, k) = \{P \subseteq TP \mid P$ is a solution for **MINL**$(\mathcal{L}(TP), S, k)\}$.*
*(4) $\mathcal{FC}(\mathcal{L}(TP), S, k, \ell) = \{P \subseteq TP \mid P$ is a solution for **FC**$(\mathcal{L}(TP), S, k, \ell)\}$.*

The following result shows that it is possible to find an element of $\mathcal{FC}$ that is at the same time an element of $\mathcal{MINL}$. This result gives us the main motivation for the present study.

Theorem 2 (Theorem 3 in [13]) *Given a finite non-empty $S \subseteq TP_\Sigma$ and $k \in N^+$. Let $\mathcal{C} = \{L \in \mathcal{L}(TP) \mid L \cap S \neq \emptyset\}$. If there exists $\ell \in N$ such that every language $L \in \mathcal{C}^k$ has a characteristic set S_L of elements of size not larger than ℓ within $\mathcal{C}^k$, then $\mathcal{FC}(\mathcal{L}(TP), S, k, \ell) \subseteq \mathcal{MINL}(\mathcal{L}(TP), S, k)$.*

Hence it suffices that we determine ℓ, a sufficient size for the elements of the characteristic sets for languages in $\mathcal{L}(TP)^k$, in order to obtain hypotheses that possess the minimality of both **FC** and **MINL**. Note that in Theorem 2, the characteristic sets required is only for within the class $\{L(p) \in \mathcal{L}(TP) \mid L(p) \cap S \neq \emptyset\}^k$. This class is considered in Section 5. We first examine the existence of these characteristic sets.

3 Existence of characteristic sets for unions of languages in $\mathcal{L}(TP)$

The following result, that shows that a characteristic set exists for every $L \in \mathcal{L}(TP)^k$ within $\mathcal{L}(TP)^k$, follows from Theorem 8 in [18] and a result discovered independently by Sato (Theorem 2.24 in [16]), Kobayashi and Yokomori (Proposition 7 in [11]).

Theorem 3 ([18] and [16, 11]) *For every $k \in N^+$, every language $L \in \mathcal{L}(TP)^k$ has a characteristic set within $\mathcal{L}(TP)^k$.*

By Theorem 3 and Theorem 2 in [13], the class $\mathcal{L}(TP)^k$ can be inferred by a learner that bases its conjecture completely on a computation of **FC**.

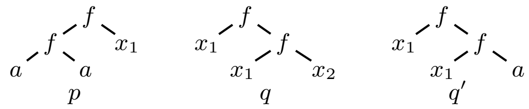
Regrettably, Theorem 3 does not hold in the case of unbounded unions. To see this, let p be a tree pattern with at least one variable, and let S be any finite subset of $L(p)$. Now S itself is a union of $card(S)$ tree pattern languages (each of a single ground tree pattern), and hence not a characteristic set for $L(p)$. One might ask if the same problem would persist if we remove the ground tree pattern languages from the class, that is, consider instead the class $\mathcal{L}(TP - TP_\Sigma)$. However, a simple analysis shows that if Σ contains any function of arity ≥ 2, then the case is no different from that which allows ground tree pattern languages. For example, let f be a function of arity 2, let $p = f(x_1, x_2)$ and let $TP' = \{f(x_1, t) \mid t \in TP_\Sigma\}$. Then every language $L \in \mathcal{L}(TP')^*$ has $L \subset L(p)$, and for every $S \subseteq L(p)$, there is a language in $\mathcal{L}(TP')^*$ which includes S. (In fact, this shows that $\mathcal{L}(TP - TP_\Sigma)^*$ is not inferable from positive examples.) Conversely, if every function in Σ is of arity ≤ 1, then trivial characteristic sets exist for languages in $\mathcal{L}(TP - TP_\Sigma)^*$.

Lemma 1. *If Σ consists only of functions of arities ≤ 1, then for any finite $P \subseteq TP - TP_\Sigma$, $S_1(P)$ is a characteristic set for $L(P)$ within $\mathcal{L}(TP - TP_\Sigma)^*$.*

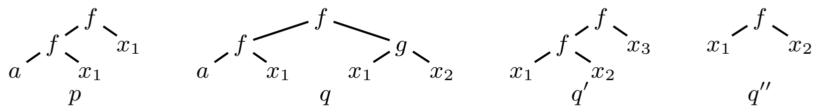
Proposition 4 *A characteristic set exists for every language in $\mathcal{L}(TP - TP_\Sigma)^*$ if and only if every function in Σ is of arity ≤ 1.*

4 Characteristic sets within unions of less than *card*(Σ) languages

We first consider intuitively the task of finding, for any two given tree patterns p and q, an element in $L(p) - L(q)$. In the following example, any tree pattern obtained from p by replacing x_1 with a function other than f cannot be refined from q. While the same cannot be done for $L(p) - L(q')$, we note that it can be done if x_1 in q' is first replaced with $f(a, a)$, replicating the functions in p.



Out of this consideration, we make the following definition. For any $p, q \in TP$, q is said to *dutifully extend* p just in case if we take each of the variable occurrences in p to be mutually independent, then p refines to q. More intuitively, q dutifully extends p just in case if a function c occurs at position ε in p, c also occurs at position ε in q. In the example below, q dutifully extends p but q' and q'' do not. Note that $q \preceq p \Rightarrow q$ dutifully extends p.



We now attempt to find a most general refinement of a tree pattern q that dutifully extends a tree pattern p. Consider the following method, the output of which we denote by mge$\{q \supseteq p\}$.

$dutifully\_extend\_p$
1. Set mge$\{q \supseteq p\}$ to q.
2. If there is any function in mge$\{q \supseteq p\}$ that conflicts with functions in p, output "failure".
3. While there exists a function node, v say, in p where no corresponding function occurs at pos(v) in mge$\{q \supseteq p\}$.
   3.1 Find a variable node u in mge$\{q \supseteq p\}$ at some prefixes of pos(v). If no such variable node exists in mge$\{q \supseteq p\}$, output "failure".
   3.2 Let t be the tree pattern at pos(u) in p. Replace each variable occurrence in t with a distinct variable not in mge$\{q \supseteq p\}$ and let t' denote the resultant tree pattern.
   3.3 Let x be the root symbol of u. Substitute each occurrence of x in mge$\{q \supseteq p\}$ with t'. If this results in any function in mge$\{q \supseteq p\}$ to conflict with functions in p, output "failure".
4. Output mge$\{q \supseteq p\}$.

It is clear that (1) mge$\{q \supseteq p\} \preceq q$, and (2) mge$\{q \supseteq p\}$ dutifully extends p. Intuitively, the method preserves the generality of mge$\{q \supseteq p\}$, by making only the necessary refinements to q to dutifully extend p (in steps 3.2∼3.3). From this, we can verify that $L(p) \cap L(q) \subseteq \text{mge}\{q \supseteq p\}$. It can also be shown that (1) there exists exactly one such $\text{mge}\{q \supseteq p\}$, unique to renaming of variables, and (2) if the method outputs "failure", then no refinement of q dutifully extends p. Since $\text{mge}\{q \supseteq p\}$ agrees with p in both structure and functions, any discrepencies between $\text{mge}\{q \supseteq p\}$ and p can involve only variables in p.

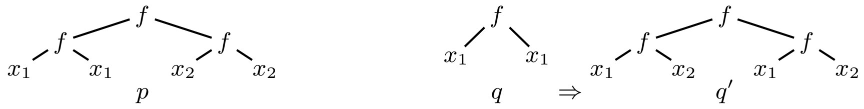
We now find a sufficient size to the elements of the characteristic sets for the unions of the tree pattern languages, when the number of languages allowed in a union is less than the alphabet size.

Lemma 5 *Let $p \in TP$ and $Q \subseteq TP$ be where $card(Q) < card(\Sigma)$. If $\{p\} \not\sqsubseteq Q$ then there exists a ground tree pattern $t \in L(p) - L(Q)$ of size $(1+\alpha)\|p\|$, where α is the largest arity of the functions in Σ.*

Proof. Without loss of generality we assume that for each $q \in Q$, either q dutifully extends p, or there exists some $q' \preceq q$ that dutifully extends p, since otherwise $L(p) \cap L(q) = \emptyset$. Let $Q' = \{\text{mge}\{q \supseteq p\} \mid q \in Q\}$. Note that $card(Q') < k$ and $L(p) \cap L(Q) = L(p) \cap L(Q')$. We now show that any such $q' \in Q'$ must be of one of two possible forms. Since q' dutifully extends p, $\|q'\| \geq \|p\|$. Now since for every $q \in Q$, $p \not\preceq q$, one of the following cases must apply for $q' \in Q'$:

(C1) A variable x occurs in p at a position where a function c occurs in q'. In this case we say that x and c witnesses that q' fulfills (C1).

(C2) A variable x occurs in p at a position where a variable y occurs in q', however y also occurs in q' at a position where a variable $x' \neq x$ occurs in p. In this case we say that x witnesses that q' fulfills (C2). In the following example, the left-most occurrence of x_1 in p witnesses that q' fulfills (C2).



This is how we construct a ground tree pattern that cannot be refined from the tree patterns that fulfill (C1). Let variable x in p (with some $c \in \Sigma$) witness that the tree patterns $Q^x \subseteq Q'$ fulfill (C1), and let $\Sigma^x \subseteq \Sigma$ be such that for each $q' \in Q^x$, there exists $c \in \Sigma^x$ that witnesses that q' fulfills (C1) as witnessed by x and c. Clearly, no $q' \in Q^x$ refines to the tree pattern obtained by replacing every occurrence of x in p with a symbol in $\Sigma - \Sigma^x$. For the tree patterns fulfilling (C2) we do the following. Let X be a collection of the variables occurring in p such that for each q' that fulfills (C2), there is a variable in X that witnesses it. If we replace each of the variables in X with a distinct function, and all other variables that occur in p but is not in X with yet another distinct function, then all the q' fulfilling (C2) cannot refine to the resultant tree pattern. The following algorithm $find\_t$ does exactly these.

We first construct a basic set of ground tree patterns of at most height 1, for use in $find\_t$. Let a be an arbitrary constant in Σ. For each function $f \in \Sigma$, let $f_a = f$ if f is a constant, otherwise f_a is tree pattern with root symbol f and the constant a as children. We let $\Lambda_a = \{f_a \mid f \in \Sigma\}$. It is clear that $card(\Lambda_a) = card(\Sigma)$.

find_t

1. Let $C = \Sigma$.
2. For each $q' \in Q'$, find a variable x in p that witnesses that q' fulfills (C1). Let X be the set of all such x found.
3. While Q' is not empty,
   3.1 For each $x \in X$, let Q^x be the set of all $q' \in Q'$ with a function occurring at some position where x occurs at in p.
   3.2 For each $q' \in Q^x$ find one function c which occurs at a position where x occurs at in p. Let C' be the collection of all such c found.
   3.3 Replace x with an element $f_a \in \Lambda_a$ where $f \in C - C'$ (this is possible because $card(C') \le card(Q') < card(C)$), and remove f from C.
   3.4 Remove Q^x from Q'.
4. Let Q'' be the set of all q' that fulfills (C2) but not (C1).
5. For each $q' \in Q''$, find a variable x in p that witnesses that q' fulfills (C2). Let X be the set of all such x found.
   5.1 If $x \in X$ has not already been replaced, replace x with an element $f_a \in \Lambda_a$ where $f \in C$, and remove f from C.
6. Replace all remaining variables in p with an element $f_a \in \Lambda_a$ where $f \in C$. (Note that at most $card(Q') < card(\Sigma)$ elements have been removed from C prior to this step.)
7. Let t be the resultant pattern. Output t.

Since there are at most $\|p\|$ variable occurrences in p, the total number of terms added to p in *find_t* is at most $\alpha\|p\|$. Since $t \not\preceq q'$ for any $q' \in Q'$, $t \notin L(q')$ for each $q' \in Q'$. Clearly, $t \in L(p)$ by the substitutions done in *find_t*, but since $t \notin L(Q')$, $t \notin L(Q)$. ∎

Theorem 6 *Let α be the largest arity of the functions in Σ and let $k < card(\Sigma)$. For every collection of less than k tree patterns $P, Q \subseteq TP$, the following three statements are equivalent:*
(i) $S_\alpha(P) \subseteq L(Q)$, (ii) $P \sqsubseteq Q$, (iii) $L(P) \subseteq L(Q)$.

Proof. (ii)⇒(iii) by definition. (iii)⇒(i) is immediate. (i)⇒(ii) by Lemma 5. ∎

The equivalence of (i) and (iii) says that for any set of less than k tree patterns $P \subseteq TP$, $S_\alpha(P)$ is a characteristic set of $L(P)$ within $\mathcal{L}(TP)^k$.

Theorem 7 *Let α be the largest arity of the functions in Σ. For every finite, non-empty $S \subseteq TP_\Sigma$ and $k < card(\Sigma)$, $\mathcal{FC}(\mathcal{L}(TP), S, k, \max(\{(1+\alpha)\|t\| \mid t \in S\})) \subseteq \mathcal{MINL}(\mathcal{L}(TP), S, k)$.*

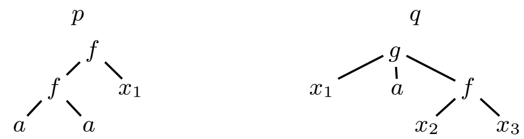
5 Characteristic sets within $\{L \in \mathcal{L}(TP) \mid L \cap S \neq \emptyset\}^k$

We now remove the alphabet size restriction placed on the number of unions allowed, and find, for any tree pattern language L, a sufficient size for constructing a characteristic set of L of only elements of up to that size, within

$\{L \in \mathcal{L}(TP) \mid L \cap S \neq \emptyset\}^k$, with the condition that there there is at least a function of arity ≥ 2 in Σ.

A node u in q *exceeds* a node v in p just in case $\mathrm{pos}(v)$ is a prefix of $\mathrm{pos}(u)$. u properly exceeds v just in case u exceeds v but v does not exceed u. A tree pattern p *exceeds* a tree pattern q just in case no leaf node in p is properly exceeded by some node in q.

Example 2 *Let $\Sigma = \{a, f, g\}$ where a is a constant, f is of arity 2, and g is of arity 3. Let $p = f(f(a, a), x_1)$ and $q = g(x_1, a, f(x_2, x_3))$. Then, p exceeds q.*



Note that the notion of "exceed" is much easier to fulfill than the notion of "dutifully extend". For any p and q where p exceeds q, $p \not\preceq q$ implies certain relationship between p and $\mathrm{mge}\{q \supseteq p\}$, which we will demonstrate in Lemma 9.

Lemma 8 *Let $p, q \in TP$ be such that no node in q properly exceeds any node of a variable in p, then $L(p) \cap L(q) \neq \emptyset \Rightarrow p$ exceeds q.*

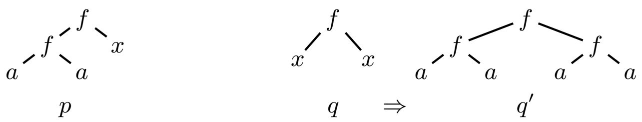
Proof. If any node in q properly exceeds any node of a constant, then clearly $L(q) \cap L(p) = \emptyset$. Hence $L(q) \cap L(p) \neq \emptyset \Rightarrow$ no node q exceeds node of constants or variables in $p \Rightarrow p$ exceeds q. ∎

Lemma 9 *Suppose Σ contains at least one function of arity ≥ 2. Let $p \in TP$ and finite set $Q \subseteq TP$ be such that (1) no node in any $q \in Q$ properly exceeds any node of a variable in p, and (2) $\{p\} \not\sqsubseteq Q$, then there exists $w \in S_d(p) - L(Q)$, where $d = \langle \|p\| + card(Q) \rangle_\Sigma$.*

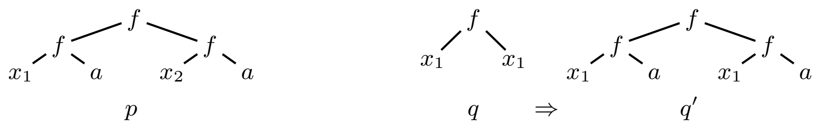
Proof. By Lemma 8, without loss of generality we assume p exceeds each $q \in Q$. Let $Q' = \{\mathrm{mge}\{q \supseteq p\} \mid q \in Q\}$. It is clear that $L(p) \cap L(Q') = \emptyset \Rightarrow L(p) \cap L(Q) = \emptyset$.

We will now show that any $q' \in Q'$ must be of one of a few forms, which will allow us to construct a $w \in L(p) - \bigcup_{q' \in Q'} L(q')$. Since p exceeds q and $p \not\preceq q$, one of the following cases must apply for $q' = \mathrm{mge}\{q \supseteq p\}$:

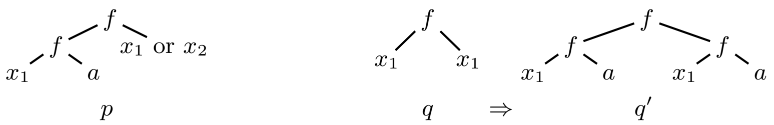
(C1) A variable x in p occurs at a position where a ground tree pattern t is found in q'. The following example shows how such a q' comes to be. Let $T_Q \subseteq TP_\Sigma$ be such that for each $q' \in Q'$ where (C1) is true, there is a $t \in T_Q$ which witnesses this.



(C2) A variable x in p occurs at a position where at the same position in q' we find a variable which has another occurrence at a position which is not occupied by x in p, as demonstrated in the following example with the variable x_2 in p.



(C3) A variable x in p occurs at a position where in q' we find a variable which has another occurrence at a node extending a variable leaf in p, as in the following example with the variable x_1 in p.



Suppose there are n distinct variables in p. Let $T \subseteq TP_{\Sigma}$ be a set of n tree patterns not in T_Q, each of the same height (this is possible by Proposition 1). Hence such T consists of trees of up to size $\langle n + \mathit{card}(Q) \rangle_{\Sigma}$. Since there are at most $\|p\|$ variables in p, each $t \in T$ is of size at most $\langle \|p\| + \mathit{card}(Q) \rangle_{\Sigma}$.

Let w be a tree obtained by replacing each distinct variable in p with a different tree in T. We claim that $w \not\preceq q'$. The case of (C1) is evident from that the variable corresponding to some $t \in T_Q$ is replaced with a $t' \notin T_Q$. The case of (C2) is evident from that every variable in p has been replaced with a different tree pattern. The case of (C3) is from that w is structurally impossible for q' to refine to, as observed by the height of the tree patterns required in variable substitutions. Hence no q' refines to w. Since $w \notin L(q')$ for all $q' \in Q'$, we have $w \notin L(Q') \Rightarrow w \notin L(Q)$. ∎

The next Theorem uses Lemma 9 to show how to find, given any $p \in TP$ and $Q \subseteq TP$ where $L(p) - L(Q) \neq \emptyset$, a finite set $S \subseteq L(p)$ where $S \not\subseteq L(Q)$.

Theorem 10 *Suppose Σ contains at least one function of arity ≥ 2, and let α be the largest arity of the functions in Σ. Given finite, non-empty $S \subseteq TP_{\Sigma}$, $p \in TP$ and finite set $Q \subseteq \{t \in TP \mid L(t) \cap S \neq \emptyset\}$. Let $h = \max(\{|s| \mid s \in S\})$. Then either $L(p) \subseteq L(Q)$, or there exists $w \in S_{\langle \|p\|(1+\alpha^h)+card(Q) \rangle_{\Sigma}}(p) - L(Q)$.*

Proof. Let Λ be the set of all constants, and tree patterns of height 1 with only variables as children. For any pattern p, let $\Theta(p)$ denote the set of all the unique tree patterns $\{p\theta \mid \theta$ is a substitution that maps each variable in p with a $t \in \Lambda\}$.

One can show that:

- (A) $L(p) = L(\Theta(p))$.

and verify that:

- (B) For any tree pattern t let $\mathrm{minvardepth}(t) = \min(\{|\mathrm{pos}(x)| \mid x$ is a variable node in $t\})$, then for all tree pattern $p' \in \Theta(p)$, $\mathrm{minvardepth}(p') = \mathrm{minvardepth}(p) + 1$. That is, intuitively, the depth of each variable node increments by 1.

Consider the following computation which, given a tree pattern p and a set of tree patterns Q, decides if $L(p) \subseteq L(Q)$.

stage 0:
Let $G_0 = \{p\}$. Go to stage 1.

stage l:
If there exists some ground tree pattern in G_{l-1} not refinable from some $q \in Q$, output false.
Remove from G_{l-1} the patterns that can be refined from some $q \in Q$.
If $G_{l-1} = \emptyset$, output true.
Let $G_l = \bigcup_{p' \in G_{l-1}} \Theta(p')$, and go to stage $l + 1$.

It is clear that if the computation halts at any stage and outputs false, $L(p) \not\subseteq L(Q)$. By Condition (A), if the computation halts at any stage and outputs true, $L(p) \subseteq L(Q)$. Note that:

- (C) At each level l, each variable in every tree pattern $p' \in G_l$ is at depth at least l (by Condition (B)).

Assume that the computation continues pass stage h. By Condition (C), at stage h, every pattern in G_h has variables at depth $\geq h$ (that is, exactly h in the case that minvardepth$(p) = 0$). On the other hand, for each $q \in Q$, $L(q) \cap S \neq \emptyset$ $\Rightarrow$ no node in q can be at a depth larger than h. Hence every variable node for any $p' \in G_h$ must exceed its corresponding node in any $q \in Q$ (if such corresponding nodes exist), and fulfills condition (1) of Lemma 9.

Let $p' \in G_h$ such that $\{p'\} \not\sqsubseteq Q$ (such p' exists by our assumption that the computation passes stage h). By Lemma 9, there exists $w \in S_d(p') - L(q)$ where $d = \langle \|p'\| + card(Q) \rangle_\Sigma$. Since there are at most $\|p\|$ variables in p, $\|p'\|$ is at most $\|p\|(1 + \alpha^h)$. $\blacksquare$

Corollary 1 *Suppose Σ contains at least one function of arity ≥ 2, and let α be the largest arity of the functions in Σ. Given finite non-empty $S \subseteq TP_\Sigma$, $k \in N^+$ and $p \in TP$. Let $h = \max(\{|s| \mid s \in S\})$. Then $S_{\langle \|p\|(1+\alpha^h)+k \rangle_\Sigma}(p)$ is a characteristic set for $L(p)$ within $\{L \in \mathcal{L}(TP) \mid L \cap S \neq \emptyset\}^k$.*

This gives us our main result.

Theorem 11 *Suppose Σ contains at least one function of arity ≥ 2. Given finite non-empty $S \subseteq TP_\Sigma$ and $k \in N^+$. Let $h = \max(\{|t| \mid t \in S\})$, $s = \max(\{\|t\| \mid t \in S\})$, and let α be the largest arity of the functions in Σ. Then $\mathcal{FC}(\mathcal{L}(TP), S, k, s(1 + d)) \subseteq \mathcal{MINL}(\mathcal{L}(TP), S, k)$ where $d = \langle s(1 + \alpha^h) + k \rangle_\Sigma$.*

6 Conclusions

We have seen how hypotheses fulfilling both the **MINL** and **FC** criteria of minimality can be obtained for the bounded unions of the tree pattern languages.

It remains to be seen whether such hypotheses can be obtained efficiently. One of the ways to achieve this is to generate a set of candidate (unions of) tree patterns (note that such a candidate set is finite since for any finite set of examples, there are only finitely many tree pattern languages that intersect non-emptily with it), and then count the number of elements in these languages of up to the size required in Theorem 7 (or 11, if the number of unions is less than the alphabet size). Such countings, if computable efficiently, will help in the search for the minimal hypotheses discussed.

Acknowledgements. The authors would like to thank the anonymous referees for corrections and valuable comments, and Prof. Kouichi Hirata for many involved discussions. Yen Kaow Ng is supported by the Japanese Government Scholarship of the Ministry of Education, Science, Sports, Culture and Technology of Japan.

References

1. D. Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.
2. D. Angluin. Inference of reversible languages. *J. of the ACM*, 29:741–765, 1982.
3. H. Arimura, H. Ishizaka, and T. Shinohara. Learning unions of tree patterns using queries. *Theoretical Computer Science*, 185(1):47–62, 1997.
4. H. Arimura, T. Shinohara, and S. Otsuki. A polynomial time algorithm for finding finite unions of tree pattern languages. In *Proc. of Nonmonotonic and Inductive Logic, Second Int. Workshop*, volume 659 of *LNAI*, pages 118–131. Springer-Verlag, 1991.
5. H. Arimura, T. Shinohara, and S. Otsuki. Polynomial time inference of unions of two tree pattern languages. *IEICE Transactions on Information and Systems*, E75-D(7):426–434, 1992.
6. H. Arimura, T. Shinohara, and S. Otsuki. Finding minimal generalizations for unions of pattern languages and its application to inductive inference from positive data. In *Proc. of STACS 94, 11th Annual Symposium on Theoretical Aspects of Computer Science*, volume 775 of *LNCS*, pages 649–660. Springer-Verlag, 1994.
7. H. Arimura, T. Shinohara, S. Otsuki, and H. Ishizaka. A generalization of the least general generalization. *Machine Intelligence 13*, 13:59–85, 1994.
8. C. Chan, M. Garofalakis, and R. Rastogi. RE-tree: an efficient index structure for regular expressions. *The VLDB Journal*, 12(2):102–119, 2003.
9. E. M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
10. W3C XML Core Working Group. Extensible Markup Language (XML) 1.0 (Third Edition). W3C Recommendation, 2004.
11. S. Kobayashi and T. Yokomori. Identifiability of subspaces and homomorphic images of zero-reversible languages. In *Proc. of Algorithmic Learning Theory, 8th Int. Conf., ALT'97*, volume 1316 of *LNAI*, pages 48–61. Springer-Verlag, 1997.
12. Y. K. Ng, H. Ono, and T. Shinohara. Measuring over-generalization in the minimal multiple generalizations of biosequences. In *Proc. of Discovery Science, 8th Int. Conf., DS 2005*, volume 3735 of *LNAI*, pages 176–188. Springer-Verlag, 2005.

13. Y. K. Ng and T. Shinohara. Inferring unions of the pattern languages by the most fitting covers. In *Proc. of Algorithmic Learning Theory, 16th Int. Conf., ALT 2005*, volume 3734 of *LNAI*, pages 269–282. Springer-Verlag, 2005.
14. Y. K. Ng and T. Shinohara. Finding consensus patterns in very scarce biosequence samples from their minimal multiple generalizations. In *Proc. of Advances in Knowledge Discovery and Data Mining, 10th Pacific-Asia Conf., PAKDD 2006*, volume 3918 of *LNAI*, pages 540–545. Springer-Verlag, 2006.
15. G.D. Plotkin. A note on inductive generalization. In *Machine Intelligence*, volume 5, pages 153–163. Edinburgh University Press, 1970.
16. M. Sato. Inductive inference of formal languages. *Bulletin of Informatics and Cybernetics*, 27(1):85–106, 1995.
17. T. Shinohara. Polynomial time inference of extended regular pattern languages. In *RIMS Symposia on Software Science and Engineering, Kyoto, Japan*, volume 147 of *LNCS*, pages 115–127. Springer-Verlag, 1982.
18. K. Wright. Identification of unions of languages drawn from an identifiable class. In *Proc. of the Second Annual Workshop on Computational Learning Theory*, pages 328–333. Morgan Kaufmann, 1989.