数理解析研究所講究録
950 巻 1996 年 207-213

A Translation Procedure for Elementary Formal Systems

杉本 典子　　石坂 裕毅
Noriko Sugimoto　　Hiroki Ishizaka
九州工業大学 情報工学部 知能情報工学科
Department of Artificial Intelligence Kyushu Institute of Technology

1 Introduction

An *elementary formal system* (*EFS*, for short) [8] is suitable to generate various formal languages [3]. The EFS has been investigated from the various viewpoints [4, 11]. In particular, from the viewpoints of *translations* on formal languages, Sugimoto and Ishizaka [9, 10] have pointed out that the translation as the relationship between two sentences is represented by a restricted EFS.

On the other hand, since EFS's are regarded as logic programs [11], the derivation procedure for EFS's can be formulated, and it is corresponding to the procedure to determine whether or not a given string is in an EFS language. That is, if the refutation from the goal clause whose argument is the string can be computed by the derivation procedure, then the string is in the EFS language. Since there exists no variable in the goal, the refutation is computable.

In this paper, we deal with the translation on EFS's, and capture the translation procedure between formal languages as a derivation procedure for EFS's. In other words, we study the problem of generating target sentences from a source sentence in a translation defined by an EFS. In this case, we have to compute the refutation from the goal containing variables, since a target sentence is unknown. We need a unifier of two strings containing variables in the refutation. In general, if both of two strings contain variables, then the number of all unifiers is infinite. Hence, it follows that the refutation from such a goal can not be computed by the procedure. On the other hand, we can not define a most general unifier along the same line of argument as a first order theory, since there exist unifiers such that we can not determine which of them is more general. In order to solve this

problem, we propose a *maximally general unifier* (mgu, for short). The mgu is a unifier which generates a maximally general string in the set of strings generated by all unifiers. Although the mgu is not unique, if one of two strings is one variable and the other does not contain the variable, then their mgu is unique and computable. Hence, the refutation from the goal containing variables can be computed by the derivation procedure using the mgu instead of a unifier. Furthermore, we show that the derivation procedure is complete to generate target sentences from a source sentence in the translation defined by a restricted EFS called a simple TEFS. Simple TEFS's are more powerful than Syntax-directed translation schemes[1, 2, 5, 6], because simple TEFS's can define translations over context sensitive languages.

This paper is organized as follows: In Section 2, we present basic definitions of concepts necessary for our discussions. In Section 3, we define a maximally general unifier and a restricted derivation procedure for EFS's. Furthermore, we show that the procedure is complete for restricted EFS's and goals. In Section 4, we show that if a translation is defined by a restricted EFS, then a target sentence can be generated by the restricted derivation procedure.

2 Preliminaries

In this section, we present basic definitions according to [4, 7, 8, 11] .

Let Σ, X and Π be mutually disjoint sets. We assume that Σ and Π are finite. We refer to each element of Σ as a *constant symbol*, to each element of X as a *variable*, and to each element of Π as a *predicate symbol*. Each predicate symbol is associated with a positive integer called its *arity*. In what follows, variables are denoted by x, y, z. For a set A, we denote the set of all finite strings of symbols from A by A^*, and the set $A^* - \{\varepsilon\}$ by A^+, where ε is an empty string.

A *term* is an element of $(\Sigma \cup X)^+$. A term is said to be *ground* if it is an element of Σ^+. An *atomic formula* (*atom*, for short) is of the form $p(\pi_1, \pi_2, \ldots, \pi_n)$, where p is a predicate symbol with arity n and each π_i is a term $(1 \leq i \leq n)$. A *definite clause* (*clause*, for short) is of the form $A \leftarrow B_1, \ldots, B_n$ $(n \geq 0)$, where $A, B_1, \ldots, B_n$ are atoms. The atom A is called the *head* and the sequence $B_1, \ldots, B_n$ of atoms is called the *body* of

the clause. A *goal clause* (*goal*, for short) is of the form $\leftarrow B_1, \ldots, B_n$ $(n \geq 0)$ and the goal with $n = 0$ is called an *empty goal*. We refer to either a term, an atom or a clause as an *expression*. For an expression α, the set of all variables occurring in α is denoted by $v(\alpha)$.

An *elementary formal system* (EFS, for short) S is a triplet (Σ, Π, Γ), where Γ is a finite set of clauses. Each clause in Γ is called an *axiom* of S. Let $S = (\Sigma, \Pi, \Gamma)$ be an EFS and G be a goal. We say that G is a *goal of* S, if all predicate symbols occurring in G are elements of Π and all constant symbols occurring in G are elements of Σ.

A *substitution* θ is a (semi-group) homomorphism from $(\Sigma \cup X)^+$ to itself such that $a\theta = a$ for every $a \in \Sigma$ and the set $\{x \in X \mid x\theta \neq x\}$, denoted by $D(\theta)$, is finite. The substitution is *ground* if $x\theta$ is ground for every $x \in D(\theta)$. If $D(\theta) = \{x_1, \ldots, x_n\}$ and $x_i\theta = \tau_i$, then we denote θ by $\{x_1/\tau_1, \ldots, x_n/\tau_n\}$. For a term π, $\pi\theta$ is the term obtained from π by simultaneously replacing each occurrence of the variable x_i in π by the term τ_i $(i = 1, \ldots, n)$. For an atom $p(\pi_1, \ldots, \pi_n)$, we define $p(\pi_1, \ldots, \pi_n)\theta = p(\pi_1\theta, \ldots, \pi_n\theta)$, and for a clause $A \leftarrow B_1, \ldots, B_n$, we define $(A \leftarrow B_1, \ldots, B_n)\theta = A\theta \leftarrow B_1\theta, \ldots, B_n\theta$.

Let α and β be a pair of expressions. Then a substitution θ is a *unifier* of α and β if $\alpha\theta = \beta\theta$. The set of all unifiers θ of α and β such that $D(\theta) \subseteq v(\alpha) \cup v(\beta)$ is denoted by $U(\alpha, \beta)$.

Let $\theta = \{x_1/u_1, \ldots, x_l/u_l\}$ and $\sigma = \{y_1/v_1, \ldots, y_m/v_m\}$ be substitutions. The *composition* of θ and σ, denoted by $\theta\sigma$, is defined as follows:

$$\theta\sigma = \{x_i/u_i\sigma \mid x_i/u_i \in \theta \text{ and } x_i \neq u_i\sigma\} \cup \{y_i/v_i \in \sigma \mid y_i \notin \{x_1, \ldots, x_l\}\}.$$

We say that α is a *variant* of β if $\alpha = \beta\theta$ and $\alpha\theta' = \beta$ for some substitution θ and θ'. A *computation rule* is a rule which selects an atom from every goal clause.

Let S be an EFS, G be a goal of S, and R be a computation rule. A *derivation from* G is a (finite or infinite) sequence of triplets (G_i, C_i, θ_i) $(i = 0, 1, \ldots)$ which satisfies the following conditions:

1. G_i is a goal, θ_i is a substitution, C_i is a variant of an axiom of S, and $G_0 = G$.

2. $v(C_i) \cap v(C_j) = \emptyset$ for every i and j such that $i \neq j$, and $v(C_i) \cap v(G_i) = \emptyset$ for every i.

3. If G_i is $\leftarrow A_1, \ldots, A_k$ and A_m is the atom selected by R, then C_i is $A \leftarrow B_1, \ldots, B_q$, A and A_m is unifiable, θ_i is an element of $U(A, A_m)$ and G_{i+1} is

$$(\leftarrow A_1, \ldots, A_{m-1}, B_1, \ldots, B_q, A_{m+1}, \ldots, A_k)\theta_i.$$

The atom A_m is said to be a *selected atom* of G_i, and G_{i+1} is said to be a *resolvent* of G_i and C_i by θ_i.

For notational convenience, we denote C_i and θ_i by ϕ if G_i is an empty goal. A *refutation* is a finite derivation ending with the empty goal.

Yamamoto [11] has formulated the derivation to give a procedure accepting languages definable by EFS's. Therefore, he has been able to assume that a given goal is ground. However, we can not assume that, because our aim is to generate unknown target sentences from a source sentence by the procedure. Hence, we formalize another derivation in following sections.

3 A restricted derivation procedure

In this section, we define a maximally general unifier as a unifier which maps unified expressions to maximally general expressions. Next, we formalize a restricted derivation by using the maximally general unifiers instead of unifiers. Finally, we give the class of EFS's in which the restricted derivation procedure is complete.

Let θ_1 and θ_2 be substitutions, and π be a term. We say that θ_1 and θ_2 are equivalent with respect to π, denoted by $\theta_1 =_\pi \theta_2$, if $\pi\theta_1$ is a variant of $\pi\theta_2$.

Let π_1 and π_2 be a pair of terms and $\theta \in U(\pi_1, \pi_2)$. The substitution θ is said to be *maximally general unifier* (*mgu*, for short) of π_1 and π_2, if for each $\theta' \in U(\pi_1, \pi_2)$ which satisfies $\theta' \neq_{\pi_1} \theta$, there exists no substitution γ such that $\theta =_{\pi_1} \theta'\gamma$. The set of all mgu's of α and β is denoted by $MGU(\alpha, \beta)$.

We define a *restricted derivation* by replaceing $U(A, A_m)$ with $MGU(A, A_m)$ in the definition of the derivation. A restricted derivation ending with the empty goal is called a *restricted refutation.*

Let S be an EFS and G be a goal of S. We say that a substitution θ is an *answer* for G of S if $D(\theta) \subseteq v(G)$. We say that an answer θ for G of S is *correct* if there exists a

restricted refutation from $G\theta$ of S. A *computed answer* θ for G of S is the substitution defined as follows;

$$\theta = \{x/\pi \in \theta_1 \cdots \theta_n \mid x \in v(G)\},$$

where $\theta_1 \cdots \theta_n$ is the sequence of mgu's used in a restricted refutation from G of S.

Let S be an EFS and G be a goal of S. We define as follows:

$$CORRECT(G) = \{\theta \mid \theta \text{ is a correct answer for } G \text{ and } G\theta \text{ is ground}\},$$

$$COMPUTED(G) = \{\theta \mid \theta \text{ is a computed answer for } G \text{ and } G\theta \text{ is ground}\}.$$

In general, it is not guaranteed that $CORRECT(G) = COMPUTED(G)$ as following examples.

Example 1 *Let* $S = (\{a\}, \{p\}, \{p(ay) \leftarrow\})$ *be an EFS and* $G = \leftarrow p(x)$ *be a goal. Then,* $CORRECT(G) = \{\{x/aa\}, \{x/aaa\}, \{x/aaaa\}, \ldots\}$ *but* $COMPUTED(G) = \emptyset$, *since the computed answer for* G *is only* $\{x/ay\}$ *and* $G\{x/ay\}$ *is not ground.*

Example 2 *Let* $S = (\{a\}, \{p, q\}, \{p(xy) \leftarrow q(x), q(y),\ q(aa) \leftarrow\})$ *be an EFS and* $G = \leftarrow p(aza)$ *be a goal. Then,* $CORRECT(G) = \{\{z/aa\}\}$ *but* $COMPUTED(G) = \emptyset$. *Because, goals derived from* G *are only variants of* $\leftarrow q_1(az), q_2(a)$ *and* $\leftarrow q_1(a), q_2(za)$.

Next, we give a sufficient condition for EFS's and goals to be $CORRECT(G) = COMPUTED(G)$. We define a *translation EFS* (*TEFS*, for short) as an EFS with at least one predicate symbol with arity 2. A TEFS $S = (\Sigma, \Pi, \Gamma)$ is *simple* if the arity of each predicate symbol in Π is 2 and each axiom of S is of the form $p(\pi_1, \pi_2) \leftarrow q_1(x_1, y_1), \ldots, q_n(x_n, y_n)$, where $x_1, \ldots, x_n, y_1, \ldots, y_n$ are mutually distinct variables, and $v(\pi_1) = \{x_1, \ldots, x_n\}$ and $v(\pi_2) = \{y_1, \ldots, y_n\}$.

If S is a simple TEFS and G is a goal of the form $\leftarrow p(w, x)$ for some ground term w, then all goals derived from G is of the form $\leftarrow q_1(u_1, y_1), \ldots, q_m(u_m, v_m)$, where $u_1, \ldots, u_m$ are ground terms and $y_1, \ldots, y_m$ are mutually distinct variables. From this property of simple TEFS's, we can obtain following theorem.

Theorem 3 *Let* S *be a simple TEFS and* G *be a goal which is of the form* $\leftarrow q_1(w, x)$, *where* w *is a ground term. Then, it follows that* $CORRECT(G) = COMPUTED(G)$.

4 Translations for Simple TEFS's

In this section, we show that if a translation is defined by a simple TEFS then we can obtain target sentences from a source sentence by the restricted derivation.

A *translation over* an alphabet Σ is a subset of $\Sigma^+ \times \Sigma^+$. Let T be a translation over Σ. For $w \in \Sigma^+$, we define a set of all target sentences of w in T, denoted by $T(w)$, as follows:

$$T(w) = \{w' \in \Sigma^+ \mid (w, w') \in T\}.$$

Let $S = (\Sigma, \Pi, \Gamma)$ be a TEFS, and $p \in \Pi$ be a predicate symbol with arity 2. Then, we define

$$T(S,p) = \{(w_1, w_2) \in \Sigma^+ \times \Sigma^+ \mid \text{ there exists a refutation from } \leftarrow p(w_1, w_2) \text{ of } S\}.$$

A translation T is said to be *defined* by a TEFS S and a predicate symbol p if $T = T(S,p)$. For a translation T, if there exists a TEFS S such that $T = T(S,p)$ for some predicate symbol p, then T is said to be *definable* by TEFS's.

A success set of an EFS S is defined as a set of all ground atoms A such that there exists a refutation from $\leftarrow A$ of S. Yamamoto [11] showed that for any EFS S, the success set of S is equal to the least Herbrand model of S. Therefore, we can also define $T(S,p)$ by the least Herbrand model.

Theorem 4 *Let T be a translation defined by a simple TEFS $S = (\Sigma, \Pi, \Gamma)$ and $p \in \Pi$. For any $w \in \Sigma^+$, $w' \in T(w)$ if and only if $\{x/w'\} \in COMPUTED(\leftarrow p(w,x))$.*

Above theorem shows that we can obtain all target sentences from source sentence w by computing all refutations from the goal $\leftarrow p(w,x)$.

5 Conclusion

We have formalized a restricted derivation for EFS's using maximally general unifiers. Furthermore, we have shown that the restricted derivation is computable and completely generates target sentences from a source sentence on the translation defined by a simple TEFS. As a result, we can implement a translation system for EFS's in nearly the same

way as the derivation of traditional logic programming languages. A simple TEFS is powerful than syntax-directed translation scheme [1, 2, 5, 6], since a simple TEFS can define over context sensitive languages. Therefore, the system is useful for translations over various formal languages.

If an EFS is not a simple TEFS, then we can not generate target sentences by the restricted derivation procedure, since there exist problems similar to Example 1 and Example 2. Thus, to extend the class of translations, we need another procedure. Furthermore, the problem of generating target sentences from a source sentence containing variables is open.

References

[1] A.V. Aho and J.D. Ullman. Properties of syntax directed translations. *J. Comput. System Sci.*, 3:319–334, 1969.

[2] A.V. Aho and J.D. Ullman. Syntax directed translation and the pushdown assembler. *J. Comput. System Sci.*, 3:37–56, 1969.

[3] S. Arikawa. Elementary formal systems and formal language - simple formal systems. *Mem. Fac. Sci. kyushu Univ. Ser.A 24*, pages 47–75, 1970.

[4] S. Arikawa, T. Shinohara, and A. Yamamoto. Learning elementary formal systems. *Theoret. Comput. Sci.*, pages 97–113, 1992.

[5] E.T. Irons. A syntax directed compiler for ALGOL-60. *CACM*, 4:51–55, 1961.

[6] P.M. Lewis and R.E. Stearns. Syntax-directed transduction. *J. ACM*, 15:465–488, 1986.

[7] T. Shinohara. Rich classes inferable from positive data: length-bounded elementary formal system. *Information and computation*, 108:175–186, 1994.

[8] R.M. Smullyan. Theory of formal systems. *Princeton Univ. Press*, 1961.

[9] N. Sugimoto. Properties of elementary formal systems as translation grammars. Technical report RIFIS-TR-CS 107, Kyushu Univ., 1995.

[10] N. Sugimoto and H. Ishizaka. Polynomial time learnability of deterministic right linear translations. Technical report RIFIS-TR-CS 120, Kyushu Univ., 1995.

[11] A. Yamamoto. Procedural semantics and negative information of elementary formal system. *Journal of Logic Programming*, 13:89–97, 1992.