Transformation Systems are More Economical and Informative Class Descriptions Than Formal Grammars

by

Lev Goldfarb

TR90-057, September 1990

Faculty of Computer Science
University of New Brunswick
P.O. Box 4400
Fredericton, N.B. E3B 5A3

Phone: (506) 453-4566
Fax: (506) 453-3566

Abstract. The concept of the transformation system was introduced earlier by the author as a basic part of a general model for pattern learning. In this paper, for several formal languages (of various types) the equivalent transformation systems are presented. From these examples one can draw the conclusion that the transformation systems give shorter and more informative structural class *descriptions* than the formal grammars.

Key words: Transformation systems, Formal grammars, Structural class description.

1. INTRODUCTION

In this paper I present descriptions of some known formal languages in terms of transformation systems, in order to contrast the two forms of structural class *description.* The questions related to the learning of grammars and transformation systems as well as those related to the efficiency of the corresponding on-line recognition algorithms will be considered in a separate paper.

In reference 1, section 4.2, I mentioned that the concept of formal grammar was introduced by N. Chomsky for the purposes quite different than those of pattern learning: he was interested in *the* formal grammar as a framework for describing a *single* language, while in pattern learning we are faced, in the language of syntactic pattern recognition, with the problem of "grammar inference" from a finite learning set of patterns. It was assumed by many that the concept of generative grammar is also adequate for the purposes of pattern learning. For the criticism of the latter see the above reference, as well as reference 2, Ch. 10. In essence, the inadequacy of the formal grammars framework manifest itself in the inherent ambiguity of the inference problem, as well as in the irrelevance of the form that the generative grammar takes.

The main objective of this paper is to illustrate, at a general level, that the concept of a transformation system with the parametric family of distance functions(1) is more adequate than that of the formal grammar for the purposes of class description. In what follows we shall limit ourselves to the string grammars and string transformation systems, although it will be quite clear that the similar observations extend to more general (non-linear) grammars and transformations systems.

2. STRING TRANSFORMATION SYSTEMS

By a string transformation system we shall mean a 3-tuple (O, *S*, *CR*), where O is a set of strings over a finite alphabet $A = \{a, b, c, \ldots\}$, $S = \{S_i\}_{i=1}^{m}$ is a finite set of substitution operations (each substitution operation permits to replace one specific substring, possibly from a class of substrings, by another substring, possibly from another class, and vice versa), and *CR* is a set of composition rules, i.e., rules that specify formation of new substitution operations from the existing operations (in *S*). Although the dynamic process involving modification of *the current* set *S* of substitution operations plays *the key* role in the new model for pattern learning, in this paper we shall not discuss this aspect of the model. The formal grammar model does not offer this dynamic capability and thus it is useful to understand the differences between the two formalisms at the static level, or as the *fixed class descriptions*.

In agreement with the latter, we shall view a *transformation system* as a pair (O, *S*), were O and *S* are defined as above.

By a *competing family* $F = \{\Delta_\omega\}_{\omega\in\Omega}$ *of distance functions* we shall mean the parametric family of distances each defined on O

$$\Delta_\omega : O \times O \rightarrow R_+ ,$$

where parameter ω is an m-dimensional weight vector with each weight corresponding to one of the operations in *S*,

$$\Omega = \{\ \omega = (w^1, w^2, \ldots, w^m) \mid w^i \geq 0 \quad \sum_{i=1}^{m} w^i = 1\ \}$$

and for a fixed ω distance $\Delta_\omega(o_1,o_2)$ is defined as the length of the shortest (weighted) path between strings o_1 and o_2 with each edge representing an application of one of the substitution operations to the string represented by the vertex of the path. When $w^1 = w^2 = \ldots = w^m$, distance $\Delta_\omega(o_1, o_2)$ corresponds to the smallest number of substitution necessary to transform one string into the other. Thus, Δ_ω can be viewed as a

generalization of the weighted Levenshtein, or edit, distance[3], in which the set of substitutions is expanded.

Although it is known that in general transformation systems Δ_ω cannot be computed (reference 4, p. 288), in this paper we shall not be concerned with the computability issues, since for the example presented here the known dynamic programming algorithm for computing Levenshtein distance can be adapted.

The model for pattern learning proposed in [1] suggests that any recursive class (language) can be described by means of a corresponding transformation system together with the corresponding set of weight vectors $\Omega^* \subset \Omega$. (The restriction of languages to the class of recursive languages is necessary only for the learning process.) To simplify the comparison of the two forms of pattern class descriptions - formal grammars and transformation systems - we shall assume that the latter description is given by the set $S = \{S_i\}_{i=1}^{m}$ together with the optimum weight vector $\omega^* = (w^1, w^2, ..., w^m)$, where w_i is the weight of S_i. This class description should be interpreted as follows: if the learning stage used two learning pattern sets (the set of positive patterns and the set of negative patterns) for each class C, then a string o belongs to class C if and only if Δ_ω^*-distance from it to one of the positive training patterns is zero. In fact, in all of the presented examples, we shall have

$$\forall o \in C \quad \exists o^1 \in C \ (o^1 \neq o) \qquad \Delta_\omega^*(o, o^1) = 0$$

$$\forall o \in C \quad \forall \bar{o} \notin C \qquad \Delta_\omega^*(o, \bar{o}) \neq 0$$

so that we shall not need to present the corresponding learning sets. Thus, *the above two properties of Δ_ω^* are the only ones that should be used in the examples below to check the adequacy of the class descriptions by means of the transformation systems.* As was mentioned above, we shall not consider here the learning process (involving the optimization of some weight function) by means of which the system starting from the basic set of deletion-insertion operations can evolve to that final transformation system.

In each of the next four sections we shall simply present some examples of formal grammars together with the corresponding descriptions by means of the transformation systems. As we shall observe, the latter provides not only the means for recognizing the patterns from the corresponding classes but also the means for producing the propositional class descriptions, another important advantage over the generative grammars.

3. EXAMPLES OF REGULAR GRAMMARS

In this section we consider two examples of regular languages. In all examples of formal grammars N denotes the nonterminal alphabet, T denotes the terminal alphabet, Σ denotes the starting symbol ($\Sigma \in N$, $N \cap T = \varnothing$), $R = \{R_i\}_{i=1}^{k}$ denotes the set of rewriting rules, and L(G) denotes the language generated by grammar G.

Example 1 (reference 5, p. 26)

$$G_1 = (N_1, T_1, \Sigma, R_1)$$

$$N_1 = \{\Sigma\}\ ,\quad T_1 = \{a, b\}\ ,\quad R_1: \Sigma \rightarrow a\Sigma$$

$$\Sigma \rightarrow b$$

$L(G_1) = \{b, ab, aab, aaab, ...\}$.

The corresponding transformation system is as follows (λ denotes the empty string).

S_1: $a \leftrightarrow \lambda$ $\qquad w^1 = 1/2$

S_2: $b \leftrightarrow \lambda$ $\qquad w^2 = 1/2$

S_3: $ab \leftrightarrow b$ $\qquad w^3 = 0$.

Example 2.

$$G_2 = (N_2, T_2, \Sigma, R_2)$$

$N_2 = \{\Sigma, A, B\}$, $T_2 = \{a, b\}$, R_2 : $\Sigma \rightarrow A$ $\quad \Sigma \rightarrow B$

$A \rightarrow a$ $\quad B \rightarrow b$

$A \rightarrow aA$ $\quad B \rightarrow bB$

$L(G_2) = \{a, aa, aaa, \ldots, b, bb, bbb, \ldots\}$.

The corresponding transformation system is as follows:

$S_1 : a \leftrightarrow \lambda$ $\quad w^1 = 1/2$

$S_2 : b \leftrightarrow \lambda$ $\quad w^2 = 1/2$

$S_3 : aa \leftrightarrow a$ $\quad w^3 = 0$

$S_4 : bb \leftrightarrow b$ $\quad w^4 = 0.$

4. EXAMPLES OF CONTEXT-FREE GRAMMARS

Example 3 (reference 7, exercise 1.3).

$$G_3 = (N_3, T_3, \Sigma, R_3)$$

$N_3 = \{\Sigma\}$, $T_3 = \{a, b\}$, R_3 : $\Sigma \rightarrow ab$

$\Sigma \rightarrow ba$

$\Sigma \rightarrow \Sigma\Sigma$

$\Sigma \rightarrow a\Sigma b$

$\Sigma \rightarrow b\Sigma a$

$L(G_3) = \{$the set of strings in which a occurs as many times as b$\}$.

The corresponding transformation system is as follows:

$S_1 : a \leftrightarrow \lambda$ $\quad w^1 = 1/2$

$S_2 : b \leftrightarrow \lambda$ $\quad w^2 = 1/2$

$S_3 : ab \leftrightarrow \lambda$ $\quad w^3 = 0$

$S_4 : ba \leftrightarrow \lambda$ $\quad w^4 = 0.$

Example 4 (reference 6, p. 322).

$$G_4 = (N_4, T_4, \Sigma, R_4)$$

$N_4 = \{\Sigma\}$, $T_4 = \{a, b\}$, $R_4 : \Sigma \rightarrow ab$

$\Sigma \rightarrow a\Sigma b$

$L(G_4) = \{ab, aabb, aaabbb, \ldots\}$.

The corresponding transformation system is as follows:

$S_1 : a \leftrightarrow \lambda$ $\quad w^1 = 1/2$

$S_2 : b \leftrightarrow \lambda$ $\quad w^2 = 1/2$

$S_3 : ab \leftrightarrow aabb$ $\quad w^3 = 0$

5. EXAMPLES OF CONTEXT-SENSITIVE GRAMMARS

Example 5 (reference 6, p. 322).

$$G_5 = (N_5, T_5, \Sigma, R_5)$$

$N_5 = \{\Sigma, A, B\}$, $T_5 = \{a, b, c\}$, R_5 : $\Sigma \rightarrow abc$

$\Sigma \rightarrow aAbc$

$Ab \rightarrow bA$

$Ac \rightarrow Bbcc$

$bB \rightarrow Bb$

$aB \rightarrow aaA$

$aB \rightarrow aa$

$L(G_5) = \{abc, aabbcc, aaabbbccc, \ldots\}$.

The corresponding transformation system is as follows:

$S_1 : a \leftrightarrow \lambda$	$w^1 = 1/3$	
$S_2 : b \leftrightarrow \lambda$	$w^2 = 1/3$	
$S_3 : c \leftrightarrow \lambda$	$w^3 = 1/3$	
$S_4 : aab^n cc \leftrightarrow ab^{n-1} c$	$w^4 = 0$	$(n \geq 2)$

Example 6 (reference 7, p. 51).

$$G_6 = (N_6, T_6, \Sigma, R_6)$$

$N_6 = \{\Sigma, A, B, C, D, E, F\}$, $T_6 = \{a, b, c\}$

R_6:

$\Sigma \rightarrow ACE$	$B \rightarrow a$	$D \rightarrow c$	$F \rightarrow c$
$A \rightarrow aBD$	$DC \rightarrow DD$	$DE \rightarrow DFc$	
$B \rightarrow aAC$	$CD \rightarrow CC$	$CF \rightarrow CEc$	
$A \rightarrow a$	$C \rightarrow b$	$E \rightarrow c$	

$L(G_6) = \{a^n b^n c^k \mid n \geq k \geq 1\}$.

The corresponding transformation system is as follows:

$S_1 : a \leftrightarrow \lambda$ $\quad w^1 = 1/3$

$S_2 : b \leftrightarrow \lambda$ $\quad w^2 = 1/3$

$S_3 : c \leftrightarrow \lambda$ $\quad w^3 = 1/3$

$S_4 : aab^n cc \leftrightarrow ab^{n-1} c$ $\quad w^4 = 0 \quad (n \geq 2)$

$S_5 : a^n b^n c^2 \leftrightarrow a^n b^n c$ $\quad w^5 = 0 \quad (n \geq 2)$

6. EXAMPLES OF GRAMMARS FOR MORE REALISTIC PATTERN CLASSES

More realistic pattern grammars, even context-sensitive, are distinguished by larger alphabets N, T and larger sets of rewriting rules *R*.

Example 7 (reference 5, p. 118). The normal human electrocardiogram (ECG) can be described by a string that is a concatenation of the substrings prbtb, prbtbb, and prbtbbb, in which "the waveform primitives are p, r, and t for the pulses and b for the quiescent times". A regular grammar generating concatenations of these three strings is

$$G_7 = (N_7, T_7, \Sigma, R_7)$$

$N_7 = \{ \Sigma, A, B, C, D, E, H \}$, $\quad T_7 = \{ p, r, t, b \}$

R7 : $\Sigma \rightarrow pA$ $\quad A \rightarrow rB$ $\quad b \rightarrow bC$

$C \rightarrow tD$ $\quad D \rightarrow b$ $\quad D \rightarrow bE$

$E \rightarrow b$ $\quad E \rightarrow bH$ $\quad E \rightarrow pA$

$H \rightarrow b$ $\quad H \rightarrow b\Sigma$ $\quad H \rightarrow pA$

The corresponding transformation system is as follows:

$S_1 : p \leftrightarrow \lambda$ $\quad w^1 = 1/4$

$S_2 : r \leftrightarrow \lambda$ $\quad w^2 = 1/4$

$S_3 : t \leftrightarrow \lambda$ $\quad w^3 = 1/4$

$S_4 : b \leftrightarrow \lambda$ $\quad w^4 = 1/4$

$S_5 : prtb \leftrightarrow \lambda$ $\quad w^5 = 0$

$S_6 : prtbb \leftrightarrow \lambda$ $\quad w^6 = 0$

$S_7 : prtbbb \leftrightarrow \lambda$ $\quad w^7 = 0$.

Example 8 (reference 5, p. 127). A context-free grammar for string description of submedian and telocentric chromosomes (see Fig. 1) is

$$G_8 = (N_8, T_8, \Sigma, R_8)$$

$N_8 = \{ \Sigma, \Sigma_1, \Sigma_2, A, B, C, D, E, F \}$, $\quad T_8 = \{ a, b, c, d, e \}$

R_8 : $\Sigma \rightarrow \Sigma_1$ $\quad \Sigma \rightarrow \Sigma_2$

$\Sigma_1 \rightarrow AA$ $\quad \Sigma_2 \rightarrow BA$

$A \rightarrow CA$ $\quad A \rightarrow AC$

$A \rightarrow DE$ $\quad A \rightarrow FD$

$B \rightarrow bB$ $\quad B \rightarrow Bb$

$B \rightarrow e$ $\quad C \rightarrow bC$

$C \rightarrow Cb$ $\quad C \rightarrow b$

$C \rightarrow d$ $\quad D \rightarrow bD$

$D \rightarrow Db$ $\quad D \rightarrow a$

$E \rightarrow cD$ $\quad F \rightarrow Dc$

The transformation system *for classifying the corresponding patterns* is as follows:

S_1 : $a \leftrightarrow \lambda$ $\quad w^1 = 0$

S_2 : $b \leftrightarrow \lambda$ $\quad w^2 = 0$

S_3 : $c \leftrightarrow \lambda$ $\quad w^3 = 0$

S_4 : $d \leftrightarrow \lambda$ $\quad w^4 = 1$

S_5 : $e \leftrightarrow \lambda$ $\quad w^5 = 0$.

7. A BRIEF DISCUSSION OF THE EXAMPLES

First of all, it is important to note that each grammar in the above examples has been chosen among the smallest possible. The corresponding transformation systems are all constructed in a consistent manner: to the set of basic single-letter deletion-insertion operations new macro-operations are added. In view of this, in estimating the relative size of the transformation systems these basic operations could be ignored. Although the corresponding algorithms for learning the macro-operations are not considered here, it is useful to mention that their construction involves one or several steps each of which utilizes two weight functions introduced in (1). (I plan to introduce such learning processes shortly in a separate paper.)

Perhaps, one of the most important differences between the two formalisms under examination is the following feature: formal grammars *generate* all the patterns in the pattern class starting from the special symbol Σ, while transformation systems capture only *the differences* (in the form of the substitution operations with nonzero weights) between the positive and negative patterns. Thus, it is not very surprising that the corresponding grammars are larger, more complex, and require an additional alphabet N of nonterminal symbols. What is more surprising is that, by the same reason, some of the substitution operations provide keys to the propositional class descriptions. This fact is

more apparent if the transformation system is constructed for several pattern classes rather than for a single class, as is the case in Example 8 (see also example in (1)).

In contrast to the transformation systems, for relatively large nonterminal alphabet, the size and the complexity of the class grammars increase proportionally to the size of the alphabet and the number of the pattern classes involved. This is due to the fact that in the case of a larger alphabet the number of the substitution operations that are necessary to add to the basic insertion-deletion operations is usually small (these new operations represent only some basic characteristic features of the pattern classes).

8. CONCLUDING REMARKS

From the above and other examples of pattern grammars and transformation systems one can draw the following conclusions, which should, of course, be further corroborated. For practical size alphabets of primitives, the description of the pattern classes offered by the transformation systems are more compact and informative than those given by the formal grammars. The reason for this lies in the fact that the rewriting (production) rules of the grammar must *generate all the patterns from some artificial origin* Σ , while the transformation system must supply, in addition to the standard deletion-insertion operations, only those substitution operations *that transform one pattern within the class into any one of the patterns within some subclass* of the class. (In the latter case the pattern class can be described as a "disjunction" of the corresponding subclasses.) The remarkable fact is that with the introduction of *the metric* view of the pattern class the "transformational" form of the class description becomes more economical and more informative. Surprising as this conclusion may seem, I believe, that it is not entirely unanticipated, since it is quite clear that *much less*

information is required to discriminate between the classes than to reconstruct the classes.

At the same time, from the analytical point of view the above fact is not surprising, since the proposed model combines the classical *discrete* computational model (Post, Turing) with the continuous structure, i.e., the cardinality argument makes the above fact more comprehensible. Thus, it is the symbiosis of the discrete and the continuous that appears to offer the shortest class description.

REFERENCES

1. L. Goldfarb, On the foundations of intelligent processes-I: An evolving model for pattern learning, *Pattern Recognition* 23, 595-616 (1990).

2. S. Watanabe, *Pattern Recognition: Human and Mechanical*, Wiley, New York (1985).

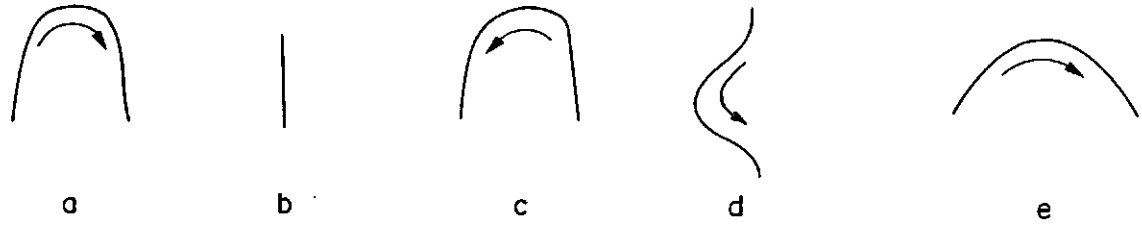
3. K. S. Fu, *Syntactic Pattern Recognition and Applications*, Prentice-Hall, Englewood Cliffs, New Jersey (1982).

4. H. R. Lewis and C.H. Papadimitriou, *Elements of the Theory of Computation*, Prentice-Hall, Englewood Cliffs, New Jersey (1981).

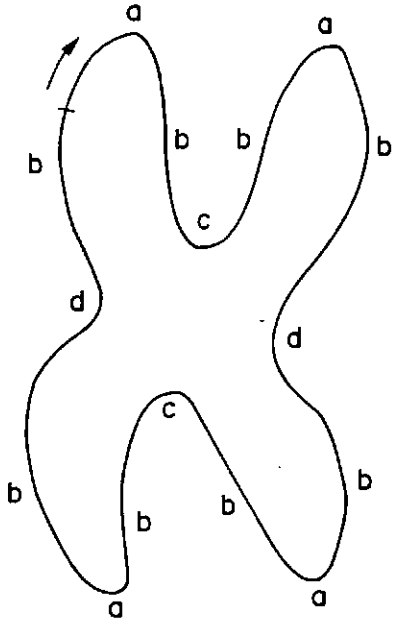
5. R. C. Gonzalez and M.G. Thomason, *Syntactic Pattern Recognition: An Introduction*, Addison-Wesley, Reading, Massachusetts (1978).

6. J. T. Tou and R. C. Gonzalez, *Pattern Recognition Principles*, Addison-Wesley, Reading, Massachusetts (1974).

7. G. E. Revesz, *Introduction to Formal Languages*, McGraw-Hill, New York (1982).

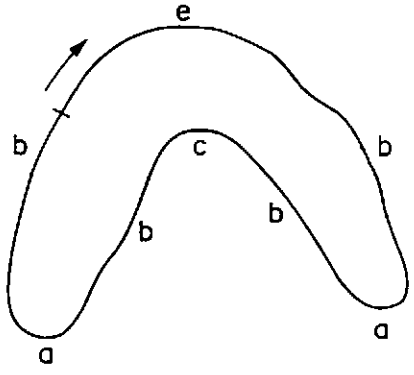


(a)



abcbabdbabcbabdb

(b)



ebabcbab

Fig. 1. (a) The primitives of the chromosome grammar. (b) The two types of chromosomes—submedian and telocentric—with the corresponding terminal sentences representing them.