Série: Monografias em Ciência da Computação
Nº 12/77

CHARACTERIZING THE REGULAR PREFIX CODES

by

Paulo A. S. Veloso

Departamento de Informática

Sērie: Monografias em Ciência da Computação
Nº 12/77

CHARACTERIZING THE REGULAR PREFIX CODES*

by

Paulo A. S. Veloso

Editor: Michael F. Challis August, 1977

* Work partially supported by the Brazilian government agency FINEP.

M3017

For copies contact

Rosane T. L. Castilho
Head, Set. de Doc. e Inf.
Deptº De Informática - PUC/RJ
Rua Marquês de São Vicente, 209 - GÁVEA
20.000 - Rio de Janeiro - RJ - BRASIL

ABSTRACT:

This paper gives an intrinsic characterization for the regular prefix codes, analogous to the one of regular sets by regular expressions. Two new binary operations on languages are introduced: "arrow" (related to star and concatenation) and "prefix-union", and the class of prefix codes is shown to be clo_ sed under these operations. A regular prefix code is expressed in terms of these operations, concatenation and single-word languages, by using the form of its minimal finite automaton.

KEY WORDS

Prefix code, regular language, regular expression, fini_ te automaton, operations on languages.

RESUMO

Êste trabalho dá uma caracterização intrínseca para os códigos de prefixos regulares, análoga à dos conjuntos regulares por expressões regulares. São introduzidas duas novas operações binárias em linguagens: "seta" (relacionada com contatenação e estrela) e "união prefixada", sob as quais a classe de códigos de prefixos é fechada. Um código de prefixos regular é expresso em termos dessas operações, concatenação e das linguagens com uma só palavra, usando a forma do seu autômato finito mínimo.

PALAVRAS CHAVES

Código de prefixos, linguagem regular, expressão regu- lar, autômato finito, operações em linguagens.

CONTENTS

1 - INTRODUCTION

The aim of this paper is to characterize the regular prefix codes, by means of their finite automata and intrinsically by analogues of regular expressions.

A prefix code is a language containing no proper prefix of its words. Such languages have been widely studied in the context of the theory of codes. They are also important in connection with finite automata, as any regular set is a finite union of languages of the form P.Q*, where P and Q are regular prefix codes (see e.g., [THIERRIN 69]), besides appearing in the study of some generalizations of the noncounting or star-free regular sets ([THIERRIN 76]).

We assume familiarity with the basic notions of finite automata (abbreviated fa's) and regular languages, and will employ the usual set-theoretical notation and consider all languages to be over a fixed alphabet. (See e.g. [GINZBURG 68 , HOPCROFT-ULLMAN 69], for more details.).

In this paper we first give some straightforward characterizations for the (regular) prefix codes. Then we introduce two new binary operations on languages, under which the class of prefix codes is closed. Finally we use the characterization for the minimal fa's and these operations to give an intrinsic characterization for the class of regular prefix codes, analogous to that of regular sets by regular expressions. Namely, the regular prefix codes are exactly the languages obtained from $\emptyset$, $\{\lambda\}$ and $\{\sigma\}$, for all $\sigma \epsilon \Sigma$, by finitely many applications of concatenation and these two new operations.

2 - PREFIX CODES AND AUTOMATA

A language P over the alphabet Σ is called a <u>prefix code</u> iff it includes no proper prefix of any of its words, i.e. $P \cap P\Sigma^{+} = \emptyset$.

The simplest prefix codes are $\emptyset$ and the single-word languages, including $\{\lambda\}$ (which is the only prefix code to which λ belongs). For a nonempty language, L, some simple equivalent definitions for prefix codes are:

- the only derivative of L with respect to any of its words is $\{\lambda\}$;
- the left quotient $L\backslash L$ of L by itself is $\{\lambda\}$;
- left concatenation by L distributes over intersection, i.e. $L.(A \cap B) = L.A \cap L.B$.
  (Recall that $w \in B\backslash A$ iff $vw \in A$ for some $v \in B$, and $D_u L = \{v \in \Sigma^{*} / uv \in L\}$).

The regular prefix codes have a rather simple characterization by means of their connected fa's as follows.

2.1 - Lemma

Let $M = \langle \Sigma, S, f, s_o, F \rangle$ be a connected fa for L. Then the following properties are equivalent.

a) L is a prefix code.
b) For every $s \in S$, $A(s) = \{w \in \Sigma^{*} / f(s,w) \in F\}$ is a prefix code.
c) No final state is reachable from any other final state by a non-null word.

Proof.

For any $s \in S$, $s = f(s_o, u)$ for some $u \in \Sigma^{*}$, and $uA(s) \subseteq L$; and $s \in F$ iff $\lambda \in A(s)$. QED

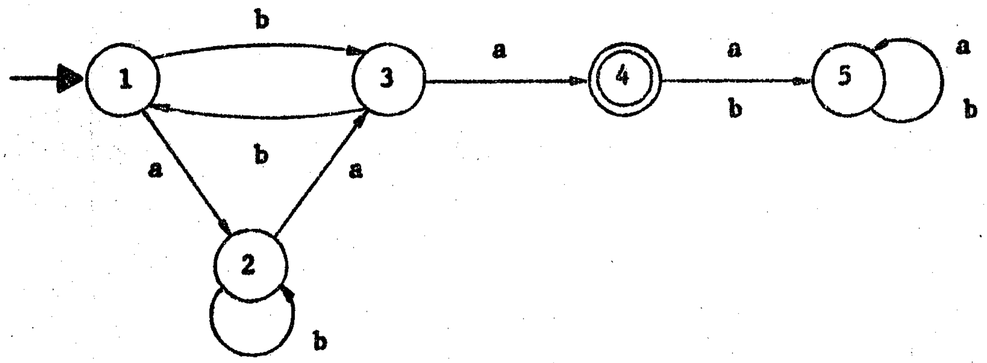
Condition(c)of 2.1 can be restated by saying that for $t \in F$ $A(t) = \{\lambda\}$, i.e. for all $w \in \Sigma^{+}$, $A[f(t,w)] = \emptyset$. Thus any two final states are equivalent, as are all of their non-null successors. These remarks yield a description of the minimal (i.e reduced and connected) fa's of the nonempty regular prefix codes, recorded here for use in section 4.

2.2 - Corollary

Let $L \neq \emptyset$ and $M=<\Sigma,S,f,s_o,F>$ be its minimal fa. Then L is a prefix code iff M has states $p \epsilon F$ and $q \epsilon (S-F)$ such that $F = \{p\}$ and for all $\sigma,\tau \in \Sigma$, $f(p,\sigma) = q = f(q,\tau)$ (i.e. q is a "sink" and p an "edge of sink").

2.3 - Example

A minimal fa with initial state 1, final state 4 and $\Sigma=\{a,b\}$ is given below by its state diagram



State 5 is a "sink" and the "edge of sink" 4 is the only final state. So, the language $(ab^*a+b)(bab^*a+b^2)^*a = (ab^*ab+b^2)^*(b+ab^*a)a$ of M is a prefix code.

3 - OPERATIONS ON PREFIX CODES

We would like to have a Kleene-like characterization for the regular prefix codes. In other words, we wish to be able to generate them by starting from very simple regular prefix codes $\emptyset$, $\{\lambda\}$ and $\{\sigma\}$, for $\sigma\in\Sigma$, and applying certain operations.

Since the class of prefix codes is clearly closed under concatenation, these is no problem in obtaining all singletons. But, this class is clearly not closed under union.

So, we shall replace set-theoretical union by the operation of prefix-union, defined by $A \# B = (A-B\Sigma^+)\cup(B-A\Sigma^+)$.

This operation is clearly commutative. It is associative as well, since $(A\#B)\#C = (A-B\Sigma^+-C\Sigma^+)\cup(B-A\Sigma^+-C\Sigma^+)\cup(C-A\Sigma^+-B\Sigma^+)$ This follows from the fact $D\Sigma^+\cup E\Sigma^+ = (D-E\Sigma^+)\Sigma^+\cup(E-D\Sigma^+)\Sigma^+$, which can be seen by taking a decomposition $w=uv$ of $w\in(D\cup E)\Sigma^+$ with $u\in(D\cup E)$ of minimal length and $v\in\Sigma^+$; for thus $u\in(D-E\Sigma^+)$ $^+$ if $u\in D$.

However, the most interesting property of the operation # for our present purposes is given in the next lemma.

3.1 - Lemma

If P and Q are prefix codes over Σ then $P\cup Q$ is a prefix code iff $P\cup Q = P\#Q$.

Proof

Notice that in this case $P\cup Q$ is a prefix code iff $P\Sigma^+\cap Q = \emptyset = Q\cap P\Sigma^+$. So, the "only if part" is clear. To see the "if part", notice that, since $P\cap Q\Sigma^+ \subseteq P\#Q$, we have $P\cap Q\Sigma^+ \subseteq Q\cap Q\Sigma^+ = \emptyset$ and similary for $Q\cap P\Sigma^+$. QED

Since the result of applying # or . to finite languages is again finite we still need another operation. It is clear that we cannot use star (for $\{\lambda\}$ is the only star-language that is a prefix code). However, we can obtain a prefix code P^*Q from prefix codes P and Q if we take adequate precautions, as the ones suggested below.

3.2 - Lemma

Let P be a prefix code over Σ and $L \subseteq \Sigma^*$ be such that $L \cap P^+L = \emptyset$. Then any non-null word $w \in P^*.L$ can be written $w = u_1 \ldots u_m v$, with $m \in \mathbb{N}$, $u_1, \ldots, u_m \in P$ and $v \in L$ in a unique way.

Proof

Suppose $w = u_1 \ldots u_m v = x_1 \ldots x_n y$, with $u_1, \ldots, u_m, x_1, \ldots, x_n \in P$ and $v, y \in L$. If $m = 0$ then $n = 0$, for otherwise $w \in L \cap P^+L$, and $v = y$. For $m > 0$ since P is a prefix code, we cannot have $|u_1| \neq |x_1|$, so $u_1 = x_1$, $u_2 \ldots u_m v = x_2 \ldots x_n y$ and the result follows by induction.

QED

Notice further that if $A \cap B\Sigma^* = \emptyset$ and $\lambda \in A^*.B$ then $A^*.B = \{\lambda\}$; for then $\lambda \in B$ and $A = \emptyset$.

These properties suggest introducing the operation arrow: $A \uparrow B = (A - B\Sigma^*)^*.(B - A^+B)$

3.3 - Proposition

If P and Q are prefix codes over Σ then so is $P \uparrow Q$.

Proof

Notice that $(P - Q\Sigma^*)$ and $(Q - P^+Q)$ satisfy the conditions of 3.2 and we may assume $\lambda \notin P \uparrow Q$. Given $w \in P \uparrow Q$, we have $w = u_1 \ldots u_m v$ with $u_1, \ldots, u_m \in (P - Q\Sigma^*)$ and $v \in (Q - P^+Q)$. Now, consider any prefix x of w. If $x \in P \uparrow Q$ then, by 3.2, we have 2 cases.

(i) Either $x = u_1 \ldots u_m y$ with $y \in Q$ a prefix of v; then $y = v$ and $x = w$.

(ii) Or, for some $i < m$, $x = u_1 \ldots u_i z$ with $z \in Q$ a prefix of u_{i+1}; but then $u_{i+1} \in (P - Q\Sigma^*) \cap Q\Sigma^*$, which is impossible! QED

Having established these closure results, we shall show in the next section that these two new operations together with concatenation suffice to generate all the regular prefix codes.

4 - INTRINSIC CHARACTERIZATIONS

We shall now characterize the class of regular prefix codes intrinsically by analogues of regular expressions. Let us first record a simple characterization for the finite prefix codes, following from the remarks of the preceding section.

4.1 - Fact

The family of finite prefix codes over Σ consists of $\emptyset$, $\{\lambda\}$ and all the finite prefix-unions of finite concatenations of single-letter languages (i.e., $\{\sigma\}$, for $\sigma\in\Sigma$).

Now consider a regular prefix code $P\neq\emptyset$. By 2.2, let $M = <\Sigma,S,f,s_o,\{p\}>$ be its minimal fa with sink q; call $S'=S-\{p,q\}$ and $S''=S'-\{s_o\}$.

We associate with states $s, t\in S$ the following prefix codes:

a) $G(s,t)$, consisting of all non-null words taking s to t for the first time, i.e.

$G(s,t) = \{w\in\Sigma^+/f(s,w) = t \ \& \ \forall u,v\in\Sigma^+(w=uv \rightarrow f(s,u)\neq t)\}$;

b) $H(s,t)$, consisting of all words taking s to t without repeating states, i.e.

$H(s,t) = \{\sigma_1\ldots\sigma_k\in\Sigma^k/f(s,\sigma_1\ldots\sigma_k) = t \ \& \ \forall\ i<j\leq k \ \ f(s,\sigma_1\ldots\sigma_i)\neq$
$\neq f(s,\sigma_1\ldots\sigma_j) \ \& \ k\in\mathbb{N}\}$.

We shall show that P can be expressed in terms of these languages using $\#$,. and $\dagger$, by a variant of a usual technique. The argument will be based on the following three claims.

4.2 - Claim

$$P = H(s_o,p) \cup \bigcup_{t\in S'} [G(s_o,t).G(t,t)^*.H(t,p)] =$$
$$= G(s_o,p)^*.H(s_o,p) \cup \bigcup_{t\in S''} [G(s_o,t).G(t,t)^*.H(t,p)]$$

Proof

Let $w=\sigma_1\ldots\sigma_k \in P$ with $|w| = k>0$. For each $j=1,\ldots, k$ call $s_k = f(s_o,\sigma_1\ldots\sigma_j)$. Notice that $s_k=p$ and for all $j<k$, $s_j \in S'$. Thus we have 2 cases.

(i) Either for all $i<j<k$, $s_i \neq s_j$; then $w\in H(s_o,p)$.

(ii) Or else, for some $i<j<k$, $s_i = s_j$. Then, let $r=\max \{j<k / \exists\, i<j \;\; s_i=s_j\}$; say $s_r=t$. Now let $j_o<j_1<\ldots<j_m=r$ be all the i's with $s_i=t$. Calling $x=\sigma_1\ldots\sigma_{j_o}$ and $z=\sigma_{r+1}\ldots\sigma_k$ we have $x\in G(s_o,t)$ and $z \in H(t,p)$. Now for $i=0,1,\ldots, m-1$, set $y_{i+1}=\sigma_{j_i+1}\ldots\sigma_{j_{i+1}}$; so $y_{i+1} \in G(t,t)$. Thus $w=x.y_1\ldots y_m.z \in$ $\in G(s_o,t).G(t,t)^*.H(t,p)$.

As the other inclusion is clear, the claim follows. QED

There are two usual techniques to compute a regular expression for the language recognized by an fa. Claim 4.2 is similar to the i-j-k method given e.g. in [HOPCROFT-ULLMAN 69, p.39]. But here we will induct on $|S|$ and the inductive step will use claim 4.3, part (b) of wich is the basic idea of the system-of-equations method as in e.g.[GINZBURG 68, p. 75].

4.3 - Claim

For all $s\in S$ and $t\in S'$

a) if $s\neq t$ then $G(s,t)$ has an fa with fewer states than M;

b) $G(t,t) = \Gamma(t) \cup \bigcup_{r\in T} \Gamma(r).G(r,t)$, where $T=S'-\{t\}$ and for $t'\in S'$ $\Gamma(t') = \{\sigma\in\Sigma / f(t,\sigma) = t'\}$

Proof

a) Transform the state graph of M by redirecting the transitions entering p or leaving t to q, removing p and reassigning initial and final states to obtain the fa $M(s,t) = <\Sigma,S-\{p\},g,s,\{t\}>$, where for any $\sigma\in\Sigma$, $g(t,\sigma)=q$ and for $r\neq t$

$$g(r,\sigma) = \begin{cases} = f(r,\sigma) & \text{if} \quad f(r,\sigma) \neq p \\ = q & \text{otherwise} \end{cases}$$

Then, by induction on $w=\sigma_1 \ldots \sigma_k \in \Sigma^k$, we have $g(s,w) = t$ iff $f(s,w)=t$ and for all $j<k$ $f(s,\sigma_1 \ldots \sigma_j) \neq t$. Thus $M(s,t)$ is a $(|S|-1)$-state fa for $G(s,t)$.

b) Clear. QED

4.4 - <u>Claim</u>

For every $t \in S'$, $G(t,t) \dagger H(t,p) = G(t,t)^* . H(t,p)$

<u>Proof</u>

Calling $A=G(t,t)$ and $B=H(t,p)$, we will show $A \cap B\Sigma^* = \emptyset = B \cap A^+ B$. First, if $u \in B$ then $f(t,u) = p$, so for any $v \in \Sigma^*$ $f(t,uv) \notin S'$, whence $uv \notin A$. Now, if $w=u_1 \ldots u_k v$ with $k>0$, $u_1, \ldots, u_k \in A$ and $v \in B$ then $f(t,u_1 \ldots u_k) = t = f(t,\lambda)$ so $w \notin B$. QED

4.5 - <u>Theorem</u>

The family of regular prefix codes over Σ is the smallest family F of languages over Σ such that

(i) F contains $\emptyset$, $\{\lambda\}$ and $\{\sigma\}$, for every $\sigma \in \Sigma$;

(ii) F is closed under prefix-union, concatenation and arrow.

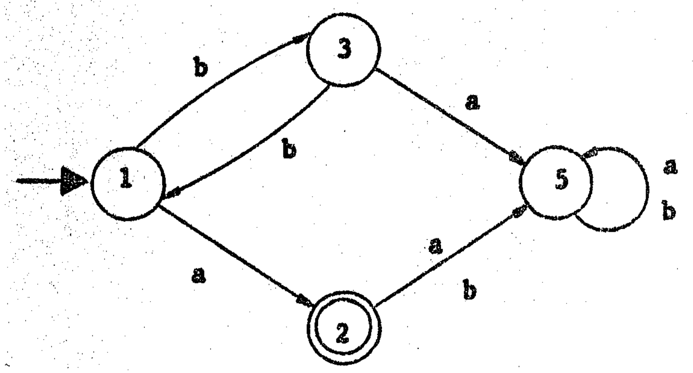
<u>Proof</u>

For $P \neq \emptyset$ with M its n-state minimal fa and the above notation, we have for every $t \in S'$ $H(s_o,p)$, $H(t,p) \in F$ (by 4.1) and $G(s_o,t)$, $G(t,t) \in F$ (by 4.3 and induction on n). Thus the result follows from 4.2, 4.4, 3.1 and 3.3. QED

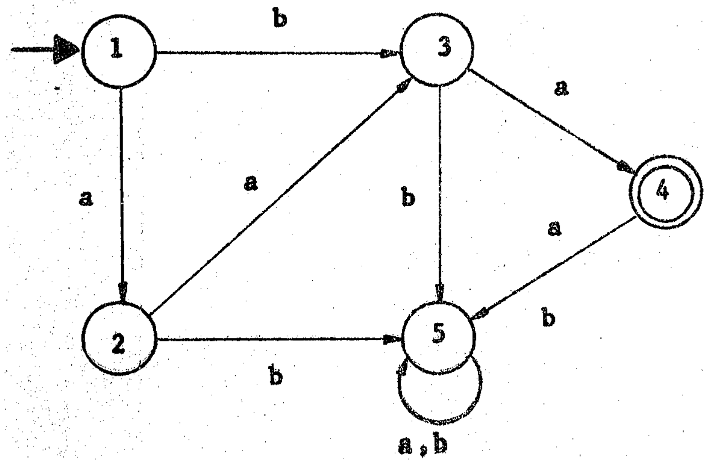
Since the proof of claim 4.3 is constructive, we have an effectiveprocedure to actually compute an F-expression for a prefix code given by its minimal fa, even though its is not a very handy one in general. We shall outline an example to illustrate the method.

4.6 - Example

Consider the fa M of example 2.3. We know that the language P recognized by M is a prefix code, and
$P = [G(1,1)\uparrow H(1,4)]\# G(1,2).[G(2,2)\uparrow H(2,4)]\# G(1,3).[G(3,3)\uparrow H(3,4)]$.
Here $G(1,1) = a\ G(2,1) \# b\ G(3,1)$; $G(2,2) = a\ G(3,2) \# b$;
and $G(3,3) = b\ G(1,3)$.
$G(1,2)$ is the language of the following 4-state fa $M(1,2)$.



Now, one can use $M(1,2)$ to determine $G(1,2) = (bb)\uparrow a$.
$H(1,4)$ is the language $(b \# aa).a$ of the fa



5 - CONCLUSIONS

We have described the regular prefix codes over the alphabet Σ in two ways. First, by characterizing their connected fa's as those where no final state is reachable from any other final state and their minimal fa's by having at most one final state which is an edge of sink. Secondly, as those languages expressable in terms of $\emptyset$, $\{\lambda\}$, $\{\sigma\}$, (for $\sigma\epsilon\Sigma$), prefix-union, con catenation and arrow.

The latter instrinsic characterization has some interesting consequences. For example, it gives nice descriptions for some classes of regular sets, such as the right simple and the right power-bounded languages of [THIERRIN 69,76].

Further, in view of the decomposition of a regular set into regular prefix codes, we see that these are, in the main , responsible for its star-height. Indeed, any regular language (resp. prefix code) can be written with one (resp. no) level of star; but, alas, many levels of arrows.

REFERENCES

. GINZBURG,A. Algebraic theory of automata. New York, Academic Press, 1968.

. HOPCROFT, J. E. & ULLMAN, J.D. Formal languages and their relation to automata. Reading, Mass., Addison-Wesley, 1969.

. THIERRIN,G. Décomposition des langages réguliers. R.I.R.O., Paris, 3e année (R3) 45-50, Sept., Oct. 1969.

. THIERRIN,G. Regular prefix codes and right power-bounded languages. Semigroup Forum, New York, 13 (1) 77-83, 1976.

Monografia em Ciência da Computação Nº 12/77
Characterising the regular prefix codes
Paulo A. S. Veloso

<u>ERRATA</u>

page 5: replace it by the attached page 5.

page 8: replace the proof of claim 4.4 by the following:

Calling $A=G(t,t)$, $B=H(t,p)$ and $A' = A^*(A/\Sigma^*)$, we will show $A' \cap B\Sigma^* = \emptyset = B \cap A^+\Sigma^*$. First, if $x\in B$ then $f(t,x)=p$, so for any $y\in\Sigma^*$ $f(t,xy) \notin S'$. Now if $u\in A^+$ then $f(t,u) = t = f(t,\lambda)$ so $uw \notin B$, for any $w\in\Sigma^*$.

3.2 - Lemma

If $P \subseteq \Sigma^*$ and $Q \subseteq \Sigma^*$ are prefix codes such that $P^+\Sigma^* \cap Q\Sigma^* = \emptyset$ then P^*Q is a prefix code.

Proof

Suppose $u_1 \ldots u_m v = x_1 \ldots x_n yz$ with $z \in \Sigma^+$; $u_1, \ldots, u_m$, $x_1, \ldots, x_n \in$ $\in (P - \{\lambda\})$ and $v, y \in Q$. As P is a prefix code, we have $u_j = x_j$ for $j = 1, \ldots, \min\{m,n\}$. We must have $m \neq n$, for otherwise $v = yz$. Now if $m < n$ then $v = x_{m+1} \ldots x_n yz \in Q \cap P^+Q\Sigma^+$, which is impossible! Also, if $m > n$ then $u_{n+1} \ldots u_m v = yz \in P^+Q \cap Q\Sigma^+$, again impossible! QED

Notice further that $\lambda \in A^*B$ iff $\lambda \in B$, and in such case $A - B\Sigma^* = \emptyset$.

These properties suggest introducing the operation arrow: $A \uparrow B = (A - B\Sigma^*)^*[B - A^*(A\Sigma^* \cup A/\Sigma^*)]$.

3.3 - Proposition

If P and Q are prefix codes over Σ then so is $P \uparrow Q$.

Proof

Notice that we way assume $\lambda \notin P \uparrow Q$ and call $R = Q - P^*(P\Sigma^* \cup P/\Sigma^*)$. To apply 3.3, assume $u_1 \ldots u_k w = yz$ with $k > 0$, $u_1, \ldots, u_k \in P$, $y \in R$ and $w, z \in \Sigma^*$. If $|w| \geq |z|$ then for some $t \in \Sigma^*$ $w = tz$ and $y = u_1 \ldots u_k t \in R \cap P^+\Sigma^*$, clearly impossible. Now, if $|w| < |z|$ then for some $i \leq k$, $x \in \Sigma^*$ and $v \in \Sigma^+$, $u_i = xv$ and $y = u_1 \ldots u_{i-1} x \in$ $\in R \cap P^*(P/\Sigma^*)$; again impossible! QED

Having established these closure results, we shall show in the next section that these two new operations together with concatenation suffice to generate all the regular prefix codes.