Six Arithmetic-like Operations on Languages

Lila Sântean

Operations on languages are intensively studied in formal language theory. For example, there are representations of some families of languages starting from simpler languages and using suitable operations. Finding of counterexamples often uses operations on languages, the theory of abstract families of languages (AFL) studies just operations, many operations appear in formal language theory applications [1], and so on.

The existing operations can be roughly clustered in three classes: set operations (union, intersection, complementation), algebraic operations (homomorphism, substitution) and purely language theoretical operations (Kleene closure, shuffle). Within this frame, it is obvious to ask for language operations corresponding to the arithmetic operations on numbers: sum, product, power, factorial, square root, and so on. Six such operations will be defined and investigated in the following, namely the compact subtraction, the literal subtraction, the generalized subtraction, the multiplication, the power, and the factorial.

The aim of this paper is to examine the closure of an abstract family of languages (when positive results are true) or directly of families in Chomsky hierarchy (when negative results hold) under these operations.

Generally, the results are the expected ones, in the sense that the family of context-sensitive languages is not closed under erasing operations, whereas for the families of context-free and regular languages, the situation is just the opposite.

1. Compact Subtraction

For a vocabulary \( \Gamma \), we denote by \( \Gamma^* \) the free monoid generated by \( \Gamma \) under the concatenation operation; the null element of \( \Gamma \) is \( \lambda \) and \( |x| \) denotes the length of the string \( x \in \Gamma^* \). The four families in Chomsky hierarchy are denoted by \( \mathcal{F}_r, r = 0, 1, 2, 3 \) (\( \mathcal{F}_0 \) denotes the family of linear languages). For other notation and terminologies in formal language theory, the reader is referred to [2].

**Definition 1.1.** Let \( L_1, L_2 \) be languages on \( \Gamma^* \). We define the compact subtraction of \( L_1 \) and \( L_2 \) by:

\[
L_1 \ominus_1 L_2 = \bigcup_{y \in \mathcal{F}_r} (x \ominus y), \quad \text{where} \quad x \ominus y = \{ x \in \Gamma^* | x = x' y \}.
\]

Compact subtraction is a generalization of right or left quotient: instead of extracting the word \( y \) from the left or right extremity of \( x \), we extract it from an arbitrary place in \( x \).

II, 1988, p. 65–75, Bruxelles, 1988
Theorem 1.1. \( \mathcal{L}_1 \) is not closed under compact subtraction.

Proof. Let \( L_1, L_2 \) be two languages on \( V^* \), we notice that
\[
\{ | \} L_2 \cap cL_1 = L_2 \setminus L_1,
\]
where \( c \) is a symbol which doesn't belong to \( V \).

As the family \( \mathcal{L}_1 \) is not closed under left quotient with regular languages, it follows that its not closed under operation \( \cap \), either.

Theorem 1.2. If \( L_1 \) and \( L_2 \) are languages on \( V^* \), \( L_2 \) regular, then there is a gsm \( g \) (with erasing) so that \( L_1 \cap L_2 = \langle g \rangle \).

Proof. Let \( A = (Q, V, \Sigma, \delta, \lambda, F, P) \) be a finite automaton that recognizes \( L_2 \). We construct the gsm:
\[
g = (Q, V, \Sigma, \delta, \lambda, F', P', F),
\]
where
\[
P' = \{ \delta(q, a) \in V \} \cup P \cup \{ \lambda(q, b) \rightarrow s \mid s \in P \}
\]
\[
\cup \{ \lambda(q, c) \rightarrow s' \mid s' \in P \} \cup \{ \lambda(q, d) \rightarrow s \mid s \in V \}
\]
\[
\cup \{ \lambda(q, e) \rightarrow s \mid s \in V, \lambda \in L_2 \}
\]
Clearly, \( g(L_2) = L_1 \cap L_2 \) and thus the proof is finished.

Corollary. \( \mathcal{L}_2, \mathcal{L}_m, \mathcal{L} \) are closed under compact subtraction with regular languages.

Open problems: The closure of the families \( \mathcal{L}_1 \) and \( \mathcal{L}_m \) under compact subtraction.

Probably, these families are not closed under compact subtraction, or, if they are, this result cannot be proved in a constructive way, because we have:

Theorem 1.3. There is no algorithm to decide whether \( L_1 \cap L_2 \) is empty or not, for \( L_1, L_2 \) arbitrary in \( \mathcal{L}_m \).

Proof. Let us consider the linear languages
\[
L_1 = \{ \sigma_1 \sigma_2 \ldots \sigma_i \sigma_i \epsilon \mid i \geq 1, i \in \{1, 2, \ldots, n \} \}
\]
\[
L_2 = \{ \epsilon \sigma_1 \sigma_2 \ldots \sigma_i \epsilon \mid i \geq 1, i \in \{1, 2, \ldots, n \} \}
\]

The statement: \( L_1 \cap L_2 \neq \emptyset \) iff there is a sequence of indexes
\[
i_1, i_2, \ldots, i_k \in \{1, 2, \ldots, n \}
\]
such that \( \sigma_1 \sigma_2 \ldots \sigma_i \sigma_i \epsilon \epsilon \) is obvious.

Therefore, we have \( L_1 \cap L_2 \neq \emptyset \) iff the POST correspondence problem has a solution, which is undecidable.

Concluding, we cannot construct in an algorithmic way a context-free grammar \( G \) so that \( L(G) = L_1 \cap L_2, L_1, L_2 \in \mathcal{L}_m \), as otherwise we could decide if \( L_1 \cap L_2 = \emptyset \) (the problem if \( L(G) \) is empty, finite or infinite is decidable for context-free grammars) — contradiction.

2. Literal subtraction

Definition. Let \( L_1, L_2 \) be languages on \( V^* \). We define the literal subtraction
\[
L_1 - L_2 = \bigcup_{x \in L_1} (x - y),
\]
where
\[
x - y = \{ x_1 x_2 \ldots x_i | x_i | y_1 y_2 \ldots y_k | x, y \mid x_1 y_1 \mid x_i y_k \mid y_j \mid k \geq 2 \}
\]
$b \in \Sigma$, $i \in [1, 2, \ldots, k - 1]$, $x_j \in \Sigma^*$, $j \in [1, 2, \ldots, k]$ (if the letters of $y$ can also be found in $x_j$ in the same order, then the literal subtraction erases them from $x_j$ without taking into account their places; else we cannot subtract $y$ from $a$).

Theorem 2.1. If $L_0$ is a regular language, then the literal subtraction $L_1 \rightarrow L_2$ can be obtained by a gsm (with erasing).

Proof. Let $A = (K, \Sigma, \delta, q_0, F)$ be a finite automaton that recognizes the language $L_0$ (therefore $F$ contains rules of the form $aa \rightarrow a$, $a \in \Sigma$).

We construct the gsm $g = (V, \Sigma, K, \delta, q_0, F)$ with $K = V$, $\delta, q_0, F$ according to $A$ and $F = F \cup \{aa \rightarrow a, a \in \Sigma\}$.

One can easily prove that $L_1 \rightarrow L_2 = g(L_0)$ (the rules of $F$ erase the symbols which come from $y$, in the correct order, and those of the form $aa \rightarrow a$ cross the symbols that will remain in $x \rightarrow y$).

Corollary. $\mathcal{F}_0, \mathcal{F}_1, \mathcal{F}_2$ are closed under literal subtraction with regular languages.

Theorem 2.2. $\mathcal{F}_1$ is not closed under literal subtraction with regular languages.

Proof. We define the gsm $g = (V, \Sigma, \delta, q_0, F, F')$, where $K = \{q_0, q_1\}$, $F = \{\delta, q_1\}$, $V = \{\sigma a \in \Sigma\}$, $F' = \{q_0, q_1 \rightarrow q_1 a \in \Sigma\}$, $\delta = \{aa \rightarrow a, a \in \Sigma\}$.

If $L \subseteq \Sigma^*$, we have the relation:

\[ g(L) = \{w \in \Sigma^* | w \not\in L\} \] (the gsm $g$ marks the symbols that are situated on the right side of the strings of $L$).

We also have the relation:

\[ L_1 / L_2 = \{g(L_1) \cap \delta(L_2)^* \} \subseteq \Sigma^* \],

where $L_1, L_2 \subseteq \Sigma^*$ and $\delta$ is a homomorphism, $k : \Sigma^* \rightarrow \Sigma^*$, $k(a) = a$.

As $\mathcal{F}_1$ is closed under intersection but is not closed under right (and left) quotient with regular languages, it follows that $\mathcal{F}_1$ is not closed under operation ---

Theorem 2.3. $\mathcal{F}_1$ and $\mathcal{F}_{1...}$ are not closed under literal subtraction with linear languages.

Proof. Let $L_1, L_2$ be the linear languages:

\[ L_1 = \{a^n b^n \} \subseteq \{a^m \} \subseteq \Sigma^* \}, \quad m \geq 1 \],

\[ L_2 = \{a^n b^n a^m \} \subseteq \Sigma^* \}, \quad m \geq 1 \].

One can easily see that:

\[ L_0 \rightarrow L_1 = \{a^n b^n | a^m b^m \} \subseteq \Sigma^* \}, \quad m \geq 1 \].

As $\mathcal{F}_2$ and $\mathcal{F}_{1...}$ are closed under intersection by regular sets but $\{a^n b^n a^m \} \subseteq \Sigma^* \}, \quad m \geq 1 \]$ is not a context-free language, it follows that these families are not closed under literal subtraction.

In fact, we have obtained a stronger result, namely that there are linear languages $L_1, L_2$ such that $L_0 \rightarrow L_1$ is not a context-free language.
3. Generalized subtraction

Definition 3.1. Let $L_1, L_2$ be languages on $V^*$. We define the generalized subtraction $L_1 \setminus L_2$ by:

\[ L_1 \setminus L_2 = \bigcup_{a \notin L_2} \{a\} \cup \{x \setminus y\}, \]

where $x \setminus y = \{x_1, \ldots, x_{n+1}|x = x_1b_1\ldots x_nb_{n+1}, y = y_1b_1\ldots y_kb_k, \text{where } y \text{ is a permutation of the word } b_1\ldots b_k, k \geq 1\}$ (if the letters of $y$ can also be found in $x$, then the generalized subtraction removes the letters of $y$ from $x$ without taking into account their places; else we cannot subtract $y$ from $x$). Notice that the generalized subtraction is a generalization of the compact and literal subtraction.

Theorem 3.1. $\mathcal{L}_2$ is not closed under generalized subtraction.
Proof. Let $L_1, L_2$ be the regular languages:

\[ L_1 = \{a^m b^m | m \geq 1\}, \]
\[ L_2 = \{a^m b^n | m \geq 0\}. \]

One can prove that

\[ (L_1 \setminus L_2) \cap \{b^* a^* f\} \neq \emptyset \text{ for } m \geq 1. \]

As $\mathcal{L}_2$ is closed under intersection by regular languages but $\{b^* a^* f\} \cap \{b^* a^* f\} = \emptyset$, it follows that $\mathcal{L}_2$ is not closed under operation $\setminus$.

Theorem 3.2. $\mathcal{L}_m, \mathcal{L}_f$ are not closed under generalized subtraction with regular languages.
Proof. Let $L_1, L_2$ be the linear languages:

\[ L_1 = \{a^m b^m | m \geq 1\}, \]
\[ L_2 = \{a^m b^n | m \geq 0\}. \]

The relation

\[ (L_1 \setminus L_2) \cap \{a^m b^m | f\} \neq \emptyset \text{ for } m \geq 1. \]

As $\mathcal{L}_m$ and $\mathcal{L}_f$ are closed under intersection by regular languages but $\{a^m b^m | m \geq 1\}$ is not context-free, it follows that $\mathcal{L}_m$ and $\mathcal{L}_f$ are not closed under generalized subtraction with regular languages.

Theorem 3.3. $\mathcal{L}_1$ is not closed under generalized subtraction with regular sets.
Proof. For each $L_1 \in \mathcal{L}_1$, there is $L_1 \in \mathcal{L}_1$, $L_1 \subseteq \{a|b|L_1 \subseteq \{a|b\}$, and for each $a \in L_1$, there is a natural $a$ such that $a|b|L_1 \subseteq L_1 \subseteq \{a|b\}$. Consider such a language $L_1 \in \mathcal{L}_1$. We have

\[ L_1 = \{L_1 \setminus a|b\} \cap V^*. \]

As $\mathcal{L}_1$ is closed under intersection, it follows that it cannot be closed under generalized subtraction with regular sets.
4. Multiplication

Definition 4.1. Let \( L_1, L_2 \) be languages on \( \Gamma^* \). We define their multiplication by:
\[
L_1 \cdot L_2 = \{ a \alpha / a \in L_1, \alpha \in L_2 \}
\]
on condition that \( \lambda \alpha = \lambda, \ a \in L_1 \)
and \( \alpha \lambda = \lambda, \ \alpha \in L_2 \).

Theorem 4.1. \( \mathcal{L}_e \) is not closed under multiplication.

Proof. Let \( L_1, L_2 \) be the regular languages
\[
L_1 = \{ \alpha \in \Gamma^* \mid n \geq 1 \},
\]
\[
L_2 = \{ \alpha \in \Gamma^* \mid \text{even} \}
\]
In accordance with definition 4.1 we have
\[
L_1 \cdot L_2 = \{ \alpha \text{even} \mid \text{even} \}
\]
which is not even context-free.

Corollary. The families \( \mathcal{L}_e \) and \( \mathcal{L}_{e\text{im}} \) are not closed under multiplication.

Theorem 4.2. \( \mathcal{L}_e \) is closed under multiplication.

Proof. A standard (straightforward, but long) construction would prove this statement; we omit the details. For a similar proof, see theorem 5.3, below.

5. Power

Definition 5.1. If \( L_1 \) and \( L_2 \) are languages on \( \Gamma^* \), we define \( L_1^{\star \star} L_2 \) (\( L_1 \text{ power } L_2 \)) by:
\[
L_1^{\star \star} L_2 = \{ \alpha_1 \alpha_2 \cdots \alpha_k / k \geq 1 \mid \alpha_i \in L_1, 1 \leq i \leq k, \alpha \in L_2 \}
\]
on condition that if \( \lambda \in L_1 \) or \( \lambda \in L_2 \), we put \( \lambda \) in \( L_1^{\star \star} L_2 \).

Theorem 5.1. \( \mathcal{L}_e \) is not closed under operation \( \star \star \).

Proof. Let \( L_1, L_2 \) be the regular languages:
\[
L_1 = \{ \alpha u \mid \text{even} \}
\]
\[
L_2 = \{ \alpha \in \Gamma^* \mid \alpha \neq \lambda \}
\]
Then, \( L_1^{\star \star} L_2 = \{ \alpha \in \Gamma^* \mid \text{even} \} \), language that is not even context-free.

Corollary. \( \mathcal{L}_e \), \( \mathcal{L}_{e\text{im}} \) are not closed under operation \( \star \star \).

Theorem 5.2. If \( L_1 \subseteq \{ \text{even} \}^* \), \( L_2 \subseteq \Gamma^* \), \( L_1 \subseteq \mathcal{L}_e \), then \( L_1^{\star \star} L_2 \subseteq \mathcal{L}_e \).

Proof. Let \( L_1, L_2 \) be two languages which satisfy the requested conditions, and \( G_1, G_2 \) the generating grammars:
\[
G_1 = (V_1, \ P_1, S_1, P_1)
\]
\[
G_2 = (V_2, \ P_2, S_2, P_2)
\]
We construct the grammar $G = (V_F, V_A, S, P)$ where $V_F = V_A = V^+_F \cup V^+_A \cup \{S, E, A, C, C', C'', C''', B, D, H, H', H'', L, F, I\}, V_F \cap V_A = \emptyset$ and $P$ is constructed as follows:

$P$ contains $P_F \cup P_A$ on condition that if $P_F$ or $P_A$ contain rules giving the null word, we eliminate them from $P_A$, and introduce it in its stead the rule $S \to \lambda$.

Moreover, we shall add to $P$ the rules (1)-(24), which will be explained in the sequel.

First, we generate $z \in L_2$, bordering it with $FC$ to its left, and with $E$ to its right:

1. $S \to FOS_E$.

Because we want to obtain $\varepsilon$ words from $L_2$, we change every letter of $z$ into $B_1$, separating them by $A$:

2. $C \to C_B A, \quad a \in V^+_F$
3. $AA \to AB_A A, \quad a \in V^+_F$
4. $AAE \to AB_A E, \quad a \in V^+_F$.

Deriving on with rules from $P_A$ we get:

$FC_{C_1 A_1 \cdots A_{C_1} B_1 \cdots B_{C_1} E_1 \cdots E_{C_1}} A_{C_2} \cdots A_{C_2} B_2 \cdots B_{C_2} E_2 \cdots E_{C_2} z_a \in L_4$.

Now, we try to obtain the word $x_{C_2}'' = y$, then $y^{(d)} = x_{C_2}'' z_a$; and so on, till we get $x_{C_2}^{(d)} z_a$.

During the first step, we work only with the first two words. Thus, we limit the working zone:

5. $C \to C_{C''}$.

$F$ marks the left extremity of the whole word, $C$ the left extremity of $x_2$, and $C''$ the other limits, in the following way: $C''$ goes to the right, crossing only the terminals, when it meets the left extremity of $x_2$, it points this out by turning itself into $C'''$ and $A$ into $B_2$; $C'''$ goes to the right; and, when it meets the right extremity of $x_2$, it turns $A$ into $D_1$, if $x_2$ is not the last word, or $E$ into $H_1$ if $x_2$ is the last word, and disappears:

6. $C'' \to a C'''$, \hspace{1cm} a \in V^+_F
7. $C'' A \to B C''$, 
8. $C''' A \to a C'''$, \hspace{1cm} a \in V^+_F
9. $C''' A \to D$
10. $C''' E \to H$

After using these rules we get either the word

$FC_{a_1 z_{C_2} \cdots a_n B_2 a_{C_2} \cdots b_n D_1} E_{C_2} z_a$, or $FC_{C_2 A_1 \cdots A_{C_2} B_2 \cdots B_{C_2} E_1 \cdots E_{C_2} z_a}$. 

\[ \frac{i_s}{i_s} \]
To obtain $x_1 \text{'}$ we have to generate a word $x_1$ for every letter of $x_2$.

We bring a letter $b$ to the left of $B$, marking it:

11. $Bb \rightarrow b'B$, $b \in V_1^\dagger$

This $b'$ goes to the left, adding a marked lining to every letter of $x_1$.

12. $ab' \rightarrow b'a''a$, $a, b \in V_1^\dagger$

13. $Ca'b' \rightarrow Ca''a'$, $a \in V_1^\dagger$, $b \in V_1^\dagger$

The marked symbols move to the left, in order, and when they attain $C$, erase it, losing their marks:

14. $ba'' \rightarrow a''b$, $a, b \in V_1^\dagger$

15. $Ca'' \rightarrow aC'$, $a \in V_1^\dagger$

After using these rules we obtain either

$F \rightarrow a_1 a_2 \ldots a_n b_1 b_2 \ldots b_n D$ or

$F \rightarrow a_1 a_2 \ldots a_n C a_2 \ldots a_1 b_1 b_2 \ldots b_n H$

We repeat these rules for every letter $b$, when we reach the last

... we destroy it:

16. $BbD \rightarrow B'A$, $b \in V_1^\dagger$

Afterwards, if $x_2$ is not the last word of $L_0$ — in the word we are
talking about —, to use the set of rules (5)—(16) again, we must bring
the current word to the initial form:

17. $aB' \rightarrow a'a$, $a \in V_1^\dagger$

18. $C'B' \rightarrow B'$

We continue the moving of $B'$ to the left, until it reaches $F$:

19. $aB'' \rightarrow a'a$, $a \in V_1^\dagger$

20. $BB'' \rightarrow FC$

Now, the current word is:

$FC \equiv a'R_0 A \ldots A x_{n-1} E_1$

and we can resume the set of rules, beginning with (5).

If $x_n$ is the last occurrence of a word of $L_0$ in the current word, then
the last $b$ disappears, and $B$ and $B'$ turn into $I$, which moves to the left,
erasing all nonterminals:

21. $BBH \rightarrow I$, $b \in V_1^\dagger$

22. $al \rightarrow Ia$, $a \in V_1^\dagger$

23. $C'I \rightarrow I$

24. $FI \rightarrow \lambda.$
From the above explanations, it easily results that $L(G) = I_1**I_2$.

To show that $L(G) \in \mathcal{F}_1$ we shall use the work-space theorem (12).

Let $z$ be a word in $L(G)$, $z \neq \lambda$, $z = \omega_1 \omega_2 \cdots \omega_k$, and a derivation $D: \omega_k \Rightarrow \omega_1 \Rightarrow \cdots \Rightarrow \omega_k = z$.

The only places $\omega \Rightarrow \omega'$ where we can have $|\omega'| < |\omega|$ are the places where we apply:

- rule (9), (16) or (18); each of them decreases $\omega$ with one letter and can be applied $|\gamma| - 1$ times in $D$.
- rule (10), (23), or (24); each of them decreases $\omega$ with one letter and can be applied once in $D$.

Consequently, we conclude that the greatest length of a word in $D$ cannot be larger than $|\omega| + 3|\gamma| - 1 + 2 + 2 + 1$.

For $k = 4$, and taking into account that the words from $I_4$ have the length greater or equal to two, we have:

$$WS(z, G) \leq \min_{\omega} WS(D, G) \leq WS(D, G) = \max_{\omega} |\omega| =$$

$$= |\omega| + 3|\gamma| - 1 + 2 + 2 + 1 = k|\omega|.

According to the work-space theorem, $L(G) = I_4**I_2 \in \mathcal{F}_1$.

Open problem. Is $\mathcal{F}_1$ closed under operation $**$?

6. Factorial

Definition 6.1. Let $L$ be a language on $V^*$. We define $L$ factorial by:

$L! = \{x! \mid x \in L\}$ where, if $x = a_1a_2 \cdots a_n$, then $a! = a_1a_2a_3 \cdots a_{n-1}$ on condition that $\lambda! = \lambda$ and $a! = a$, $a \in V$.

Theorem 6.1. $\mathcal{F}_2$ is not closed under operation $!$.

Proof. Let $L$ be the regular language $L = \{a^n \mid n \geq 1\}$. In accordance to definition 6.1, $L! = \{a^{n+1} \mid n \geq 0\}$, language which is not even context-free.

Corollary. $\mathcal{F}_0$, $\mathcal{F}_1$, are not closed under operation $!$.

Theorem 6.2. $\mathcal{F}_1$ is closed under operation $!$.

Proof. Let $L$ be a language in $\mathcal{F}_1$, and $G = (V_1, V_2, S, P)$ the generating grammar.

Let $G'$ be a grammar, $G' = (V_1', V_2, S', P')$, where $V_1' = \{S', X_1, X_2\} \cup \{v_1, V_2 \cup \{c\}\}$ and $P'$ is constructed as follows:

$P'$ contains $D$. We shall also introduce into $P'$ the rules (1)-(7) constructed in the following way:

First, we produce a word from $L$:

(1) $S' \rightarrow X_1X_2S'X_2$. 

A derivation will continue only with rules from \( O \) until we obtain
\( X_tX_t \in \mathcal{L}, t \in T \). Assuming that \( a = a_1a_2 \ldots a_n \) we'll try to produce a
lining of the first \( n - 1 \) letters. If on the right side of \( X_t \) there are at least
two letters, the first one passes on the left side of \( X_t \) and produces a mar-
ked lining:

\[
(2) \quad X_t a b \rightarrow a' a X_t b, \quad a, b \in \Gamma_T.
\]

When we attain the last letter of \( a \), which must not be doubled, we
pass it to the right side of \( X_t \), and point this out by marking \( X_t \):

\[
(3) \quad X_t a X_t \rightarrow X_t X_t a, \quad a \in \Gamma_T.
\]

All the unmarked symbols pass, in order, to the right side of \( X_t \):

\[
(4) \quad a \theta \rightarrow a, \quad a \in \Gamma_T, \quad \theta \in \mathcal{L}_\theta \cap \{X_t, X_t\}.
\]

In this moment, on the right side of \( X_t X_t \) we have the initial word,
and on the left side, the first \( n - 1 \) letters, marked.

We have to repeat the preceding operations and, with this end in
view, we move \( X_t \) to the left, until it reaches the extremity, when it
turns back into \( X_t \). In this way, \( \mathcal{L} \) erases all the marks, so that, when
it reaches the left extremity and becomes \( X_0 \), we can repeat our method
for the \( n - 1 \) letters between \( X_0 X_t \) and \( X_t \):

\[
(5) \quad a' X_t \rightarrow X_0 a', \quad a \in \Gamma_T
\]

\[
(6) \quad X_0 X_t \rightarrow X_0 X_t.
\]

Finally, when we have no more letters between \( X_0 X_t \) and \( X_t \):

\[
(7) \quad X_0 X_t \rightarrow \varepsilon.
\]

One can easily see, from the above explanations, that \( L(\mathcal{L}) = \{ \varepsilon \} \ L \).}

\( L(\mathcal{L}) \) is clearly context-sensitive.

Let \( k \) be the homomorphism \( k : (\Gamma_T \cup \{\varepsilon\})^* \rightarrow F_T \), defined by

\[ k(a) = a, \quad a \in \Gamma_T, \quad k(\varepsilon) = \lambda. \]

We have that \( k(L(\mathcal{L})) = L \).

\( L \) is closed under restricted homomorphisms, so \( k(L(\mathcal{L})) \in \mathcal{L}_T \), there-
fore \( L \in \mathcal{L}_T \). Thus, the proof is complete.

NOTES: I wish to express my gratitude to Dr. Gheorghe Păun for his
suggestions and remarks.

REFERENCES

1. G. Păun, "Mecanismi generale ale proceselor economice" (Generative Mechanisms for