COMPLETE LANGUAGE TABLES

U. Brandt and H.K.-G. Walter

*Keywords*: Languages, language tables, defect, complete tables, domino problems, decidability.

*Abstract*: A language table is a two dimensional data structure, normally a square, which is built up like a "crossword puzzle" associated to a language $L$. The paper deals with the problem to construct to a given language $L$ language tables where no zero entries (i.e., entries different from letters) occur, so called complete tables. We show for example that whether or not there exists a language table for $L$ of size $n$ for some $n$, is undecidable for regular languages even defined over a two letter alphabet, though it is decidable for standard events. The proof shows that language tables are more powerful than dominoes because we can encode tilings of the plane, squares, etc., by complete language tables of very simple languages.

1. INTRODUCTION

Language tables are two dimensional data structures which are built in "crossword puzzle" manner. That means,
given a language L over some alphabet X, we consider con-
nected squares of empty entries or letter entries such
that reading rows from left to right or columns from top
to bottom we meet either words of L or isolated letters.
Language tables therefore define patterns constituted of
various components such that horizontally and vertically
the structure is controlled by the syntactical structure
of the given language.

There are various natural questions associated with
such a device. One measure is to count the number of empty
places, the defect of a table. This measure defines in
some sense the compactness of possible patterns. The most
compact pattern is a table without any empty entries, a
so-called complete table. In this paper we study deci-
dability problems on completeness; in [2] the combinatoric-
al aspects of this measure is investigated.

We focus our interest to the following decidability
problems.

(1) The n-CTP (complete table problem): Given n, is
there a complete table for L of size n?

(2) The CTP: Is there an n, such that there exists
a complete L-table of size n > 1?

(3) The infinite CTP: Is there an infinite sequence
of complete L-tables increasing in size?
If X is a one-letter alphabet, all three problems can be rewritten easily to very well-known decidability problems. The situation is completely different for greater alphabets. Our main result is that the CTP and the infinite CTP are undecidable for regular sets, even defined over two letter alphabets, though decidable for standard events, whereas the n-CTP is NP-complete for regular sets and contextfree languages.

The method to prove these results relies heavily on the fact that "tiling" problems give rise to construct complete tables for very simple (i.e., regular) languages. We use the paper of R.M. Robinson [3], where Turing machine computations are encoded by tilings of the plane, squares, etc.

2. BASIC NOTATIONS AND PRELIMINARY CONSIDERATIONS

We assume that the reader is familiar with the basic definitions and results of formal language theory (s. for ex. [1]).

Consider an alphabet X and a symbol O \notin X, which will represent empty entries. We are dealing with (n,n)-matrices A over X U O.

If A is such a matrix, then
\[(RA)_i = A[i,1]...A[i,n]\] and \[(CA)_i = A[1,i]...A[n,i]\] for \(1 \leq i \leq n\).

Obviously, \((CA)_i = (RA^T)_i\) if \(A^T\) is the transposed matrix.

To any such matrix we associate the graph \(G_A\) with vertices 
\[\{(i,j)/1 \leq i,j \leq n\}\]
and edges 
\[\{((i,j),(k,l))/|i-k|+|j-l|=1,A[i,j] \neq 0\text{ and }A[k,l] \neq 0\}\].

Now, let \(L \subseteq X^*\) be a language. We assume \(X \subseteq L\).

**Definition:**

1. \(A\) is an R-L-table (of size \(n\)) if and only if \(G_A\) is connected and 
\[(RA)_i \in O^* \cdot (L \cdot O^\dagger) \cdot O^* \text{ for } 1 \leq i \leq n.\]

2. \(A\) is a C-L-table (of size \(n\)) if and only if \(A^T\) is an R-L-table.

3. \(A\) is an L-table (of size \(n\)) if and only if \(A\) is both an R-L-table and a C-L-table.

**Example:** Consider the Dyck-language \(D_1 \subseteq \{(),\}^*\). Let 
\[L = D_1 \cup \{(),\} \]. Then 
\[A_1 = \begin{bmatrix} 0 & 0 & ( & 0 & ( & ) \\ 0 & ( & ) & ( & ) & 0 \\ ( & ) & ( & ) & ( & ) \\ 0 & ( & ) & ( & ) & 0 \\ ( & ) & ( & ) & ( & ) \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad A_2 = \begin{bmatrix} ( & ) & ( & ) & ( & ) \\ ( & ) & 0 & ( & 0 & 0 \\ ) & 0 & 0 & ) & 0 & ( \\ ) & 0 & ( & ) & ( & ) \\ ( & ) & 0 & ( & ) & 0 \\ ) & 0 & ( & ) & 0 \end{bmatrix}\]
are both \( L \)-tables.

We are interested in filling tables in the best possible way. Therefore we define for \( w \in (X \cup \{0\})^\ast \),
\[
|w|_0 = \text{number of occurrences of } 0 \text{ in } w,
|w| = \text{length of } w
\]
and
\[
def(A) = \prod_{i=1}^{n} |(RA)_i|_0.
\]

We call \( A \) complete if \( \text{def}(A) = 0 \).

Let us start considering the case \( X = \{a\} \). If there is a word \( w \) in \( L \subseteq \{a\}^\ast \) then, obviously, there is a complete \( L \)-table of size \( |w| \) \( (A[i,j] = a \text{ for } 1 \leq i,j \leq \leq |w|) \).

Hence we can conclude the following facts:

1. To a given \( n \) there exists a complete \( L \)-table \( A \) of size \( n \) if and only if there is a word \( w \in L \) with \( |w| = n \).

2. There exists a complete \( L \)-table \( A \) of size \( n > 1 \) if and only if \( L \cap \{w/|w| = n\} \neq \emptyset \).

3. There exists an infinite sequence \( A_1, A_2, \ldots \) of complete \( L \)-tables \( A_i \) of size \( n_i \) with \( n_1 < n_2 < \ldots \) if and only if \( L \) is infinite.

Thus our three decidability problems are completely reduced to the very well-known decidability problems for
languages $L \subseteq \{a\}^\ast$.

Now consider $L \subseteq X^\ast$ for an arbitrary $X$. Then the n-CTP is decidable for $L$ if the question "$w \in L?$" is decidable for every $w \in X^\ast$ with $|w| = n$. For we can determine in this case

$$L_n = L \cap \{w/|w| = n\}$$

and check whether or not a complete $L_n$-table of size $n$ exists. Obviously, the n-CTP for $L$ is in NP if there is a recognition algorithm for $L$ of polynomial time complexity.

Before we start to prove our main results, we show that the CTP and the infinite CTP are decidable for standard events.

**Lemma 1.** If $L = S_{\alpha\gamma_w}$ is a standard event, then there is a complete $L$-table of size $n+1$ if and only if there are two words $w, v$ and a letter $x$ with $wx \in \alpha^+ \cap L$, $xv \in \omega^+ \cap L$ and $|w| = |v| = n$.

**Proof.** Suppose there is a complete $L$-table $A$ of size $n+1$. Consider $(RA)_1 = w \cdot A[1,n+1]$ and $(CA)_{n+1} = A[1,n+1] \cdot v$. Using $x = A[1,n+1]$, we get immediately the result.

On the other hand, let $wxv$ be given according to the assumption. Let $x = x_1 \ldots x_n$, $v = y_1 \ldots y_n$, then build
the following table

\[
\begin{bmatrix}
  x_1 & x_2 & \ldots & x_n & x \\
  x_2 & x_3 & \ldots & x & y_1 \\
  \vdots & \vdots & & \vdots & \vdots \\
  x_n & x & \ldots & y_{n-2} & y_{n-1} \\
  x & y_1 & \ldots & y_{n-1} & y_n
\end{bmatrix}
\]

which is obviously a complete L-table.

Since it is decidable, whether or not this kind of word \(wxv\) exists in \(L\) for every standard event \(L\), we get the following

**COROLLARY.** The CTP is decidable for standard events.

In the following denote by \(k_L\) the constant of the pumping lemma for the regular set \(L\) (for example, derived from the minimal acceptor).

**LEMMA 2.** If \(L = S_{\alpha\gamma\omega}\) is a standard event then the following two statements are equivalent:

(i) There is a sequence \((A_i)_{i=0}^\infty\) of complete \(L\)-tables of increasing size \(n_i\).

(ii) There exist two words \(w, v\) and a letter \(x\) with \(wx \in \alpha^+ \cap L, xv \in \omega^+ \cap L, |wx| > k_L\) and \(|xv| > k_L\).

**Proof.** (i) \(\rightarrow\) (ii): By Lemma 1 we can assign to every \(i\), words \(w_i, v_i\) and a letter \(x_i\) such that \(w_i x_i \in \alpha^+ \cap \omega^+ \cap L\) and \(x_i v_i \in \omega^+ \cap L\) and \(|w_i x_i| = |x_i v_i| = n_i\). Thus
(ii) follows immediately choosing \( n_1 > k_L \).

(ii) \( \Rightarrow \) (i): Let \( w, v, x \) be given according to the assumption. Then, by the pumping lemma for regular sets:

1. \( wx = w_1^1 w_2^2 w_3^3, |w_2| > 0 \) and \( \forall n \geq 1: w_1^1 w_2^2 w_3^3 \in L \),
2. \( xv = v_1^1 v_2^2 v_3^3, |v_2| > 0 \) and \( \forall n \geq 1: v_1^1 v_2^2 v_3^3 \in L \).

Consider \( i \in \mathbb{N} \). Then there exist \( j, l \geq 1 \) with

\[ |w_1^j w_2^l w_3^3| \geq i \quad \text{and} \quad |w_1^j w_2^l w_3^3| \leq |v_1^1 v_2^2 v_3^3|. \]

We get \( v_1^1 v_2^2 v_3^3 = xv' \) for some appropriate \( v' \) and \( xv' \in \omega^+ \cap L \) because

\( xv = v_1^1 v_2^2 v_3^3 \in \omega^+ \cap L \).

Furthermore, \( xv'' \in \omega^+ \cap L \) for every \( v'' \) with \( v''u = v' \) for some \( u \). Choose \( v'' \) such that \( |xv''| = |w_1^j w_2^l w_3^3| \). Analogously, \( w_1^j w_2^l w_3^3 = w'x \) for some \( w' \) and \( w'x \in \alpha^+ \cap L \).

Now, replacing in the construction of Lemma 1 the words \( w \) and \( v \) by the word \( w' \) and \( v'' \), we get a complete \( L \)-table of size \( n_1 \geq i \). Since there exists for every \( i \), a complete \( L \)-table of size \( n_1 \geq i \), statement (i) follows immediately.

Since it is decidable whether or not \( L \) fulfills assumption (ii) for every standard event \( L \), we get the following
COROLLARY. The infinite CTP is decidable for standard events.

3. UNDECIDABILITY RESULTS

We want to show that the CTP and the infinite CTP are both undecidable for regular sets. To do this, we connect both problems with domino problems of tiling the plane respectively squares of arbitrary size. Following R.M. Robinson [3], we can encode Turing machine computations by "tiling" the plane with an associated set of dominoes. We don’t need the construction of this domino set but the reader can visualize our method considering the appropriate tiling problems. We sharpen the general undecidability result for regular sets over $X = \{a, b\}$.

Our proof includes three parts:

I. Instead of using a single language $L$ for both, rows and columns, we use a pair of languages $(L_1, L_2)$ controlling separately columns and rows.

II. We encode Turing machine computations by tables, where rows and columns are controlled by two languages $L_1$ and $L_2$ which are essentially standard events. Here the method of R.M. Robinson is used.

III. We encode complete tables over arbitrary alphabets to complete tables over the two letter alphabet $\{a, b\}$. 
Let us start with the first step. We extend the definition of an L-table to pairs \((L_1, L_2)\) of languages: an \((n, n)\)-matrix \(A\) (over \(X \cup \emptyset\)) is an \((L_1, L_2)\)-table if and only if one of the following two conditions holds:

(i) \(A\) is an \(R-L_1\)-table and a \(C-L_2\)-table, or

(ii) \(A\) is an \(R-L_2\)-table and a \(C-L_1\)-table.

Now, we show that we can merge \(L_1\) and \(L_2\) together to one language \(L\) preserving regularity and completeness. This is done by an endmarker technics. Consider a set of letters

\[ H = \{ \#_1, \#_2, \#_3, \#_4, \$, \emptyset \} \]

with \(H \cap X = \emptyset\). Let \(X' = X \cup H\).

Define \(L\) by

\[ L = \#_1\$, \#_2 \cup \#_3 \$, \#_4 \cup \$L_1\$ \cup \#_1 \emptyset \cup \#_3 \emptyset \cup \#_2 \emptyset \cup \#_4 \emptyset \cup \emptyset \emptyset \emptyset \].

Observe that \(L\) is regular if and only if \(L_1\) and \(L_2\) are both regular. Obviously, any complete \((L_1, L_2)\)-table of size \(n\) transfers to a complete \(L\)-table of size \(n+2\).

Now, we show the converse correspondence, namely, that every complete \(L\)-table of size \(n > 2\) can be transformed into a complete \((L_1, L_2)\)-table. Consider a complete \(L\)-table \(A\) of size \(n > 2\). All words in \(L\) must start with \(#_1, #_2, #_3, $\) or \(\emptyset\). Assume \(A[1, 1] \in \{$, \emptyset\}\). Then \(A[1, 2]=0\)
or \( A[1,2] \in X \) and therefore \( A[2,2] = 0 \).

Now, assume \( A[1,1] \in \{ \#_2, \#_3 \} \). Then \( A[1,j] = \#_4 \) for some \( 1 < j \leq n \). Hence \( A[2,j] = 0 \). In conclusion \( A[1,1] = \#_1 \) because \( \text{def}(A) = 0 \). Thus

\[
(RA)_1 \in \#_1^{\#_2} \cap \#_1 \not\subseteq \#_3.
\]

*First case:* Let \((RA)_1 \in \#_1^{\#_2}\). Then

\[
(CA)_1 \in \#_1^{\#_2} \cap \#_1 \not\subseteq \#_3.
\]

Assume \((CA)_1 \in \#_1^{\#_2}\). Then we get \((RA)_n \in \#_2 \not\subseteq \#_4\), i.e.,

\[
(CA)_2 = \#w \not\subseteq \# \text{ for some } w \in X' \cup 0.
\]

Since there is no word in \( L \) starting with \$ and terminating with \( \not\subseteq \), \( A[j,2] = 0 \) for some \( 1 < j < n \), and therefore \( \text{def}(A) > 0 \) — a contradiction. Thus we get \((CA)_1 \in \#_1 \not\subseteq \#_3\), i.e., \((RA)_n \in \#_3 \not\subseteq \#_4\) and \((CA)_n \in \#_2 \not\subseteq \#_4\). Since \( \text{def}(A) = 0 \),

\[
(CA)_l \in \#L_1 \$ \text{ and } (RA)_l \not\subseteq L_2 \not\subseteq \# \text{ for } 1 < l < n.
\]

Hence by deleting the borderlines we obtain an \((L_1,L_2)\)-table \( A' \) of size \( n-2 \) with \( \text{def}(A') = 0 \).

The *second case*, \((RA)_1 \in \#_1 \not\subseteq \#_3\), is symmetric.

Thus we have proven

**Lemma 1.** To any pair of languages \( L_1,L_2 \subseteq X^* \) we can construct a language \( L \) such that the following statements are equivalent for any \( n > 2 \):

1. There exists a complete \((L_1,L_2)\)-table of size \( n \).
2. There exists a complete \( L \)-table of size \( n-2 \).
Furthermore, $L$ is regular if and only if $L_1$ and $L_2$ are both regular.

The next step in our proof is the encoding of Turing machine computations by language tables.

We follow R.M. Robinson. Consider a one tape Turing machine $T$ which moves in every step and stops with an accepting state, otherwise it doesn't stop (s. [3]). First we describe the alphabet and then the two languages $L_1, L_2$, which control the desired table in such a way that a computation is encoded.

Let $B_T$ denote the alphabet, $Q_T$ the set of states and $A_T$ the (deterministic) program of $T$. Furthermore we indicate the initial state by $\text{in}(T)$, the "accepting" state by $\text{stop}(T)$ and the empty-cell symbol by $\emptyset$. We assume $\text{in}(T) \neq \text{stop}(T)$. Now, the alphabet of $L_1$ and $L_2$ consists of three parts $L, M, A$ representing $L$: "letter tiles"
$M$: "merging tiles"

$$
\begin{array}{c}
\begin{array}{c}
q \\
ql
\end{array}
\end{array} \quad \begin{array}{c}
\begin{array}{c}
a \\
aq
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
q \\
ql
\end{array}
\end{array} \quad \begin{array}{c}
\begin{array}{c}
a \\
aq
\end{array}
\end{array}
\end{array}
$$

$q \in Q_T$, $a \in B_T$

$A$: "action tiles"

$$
\begin{array}{c}
\begin{array}{c}
q' \\
qa
\end{array}
\end{array} \quad \begin{array}{c}
\begin{array}{c}
a' \\
aq
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
q' \\
qa
\end{array}
\end{array} \quad \begin{array}{c}
\begin{array}{c}
a' \\
aq
\end{array}
\end{array}
\end{array}
$$

$qaa'q'R \in \Delta_T \quad qaa'q'L \in \Delta_T$

Let the functions left, right, top and bottom from $L \cup M \cup A$ into $(Q_T \cup B_T)^*$ determine the word on the left, right, top and bottom of every tile.

The design of the language pair encoding computations can be derived by the following picture

```
+ tape +

<table>
<thead>
<tr>
<th>time</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>A</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>A</td>
<td>M</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>A</td>
<td>M</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>A</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>A</td>
<td>L</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>L</td>
<td>M</td>
<td>A</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
</tr>
</tbody>
</table>
```
By this picture the "tape" language $L_1$ is given by
$$L_1 = L^* \cdot \{ x y / ((x \in M \land y \in A) \lor (x \in A \land y \in M)) \land \text{right}(x) = \text{left}(y) \} \cdot L^*$$

The "time" language $L_2$ will be designed as a standard event $L_2 = \Sigma_{\alpha \gamma \omega}$. For all decision problems $\alpha$ and $\gamma$ remain unchanged whereas $\omega$ is adjusted appropriately.

Remark: Obviously, with little modifications $L_1$ can be designed as a standard event, too.

Let
$$\alpha = \{ x / (x \in L \cup M \land \text{top}(x) = \emptyset) \lor (x \in A \land \text{top}(x) = \text{in}(T) \emptyset) \}$$
and
$$\gamma = \{ (x, y) / ((x \in L \land y \in M) \lor (x \in M \land y \in A) \lor (x \in A \land y \in L)) \land \text{bottom}(x) = \text{top}(y) \} .$$

To show the undecidability of the CTP for regular sets we define $\omega$ in such a way that the Turing machine $T$ stops when started with the empty tape if and only if there exists a complete $(L_1, L_2)$-table of size $n > 1$. Then the result follows immediately by Lemma 1.

Let
$$\omega = \{ x / x \in L \cup M \lor (x \in A \land \text{bottom}(x) = \text{stop}(T)a \text{ for some } a \in E_T) \} .$$
Consider a complete $(L_1, L_2)$-table of size $n > 1$, where
\( L_2 = \Sigma_{\alpha\gamma\omega} \). Then \((RA)_i\) determines the \(i\)-th configuration of \(T\) when started with the empty tape. Furthermore, 
\( \text{bottom}(\lambda[n, l]) = \text{stop}(T)a \) for some \( a \in B_T \) and some \( 1 \leq l \leq n \). Thus \(T\) stops. On the other hand, every accepting computation of \(T\) when started with the empty tape yields a complete \((L_1, L_2)\)-table of size \(n > 1\) (remind in(T) \(\neq\) stop(T)!) because we can encode every configuration of \(T\) to a word of \(L_1\) and control the computation by \(L_2\). Extending the configurations by an appropriate number of \(\emptyset\)-symbols, we achieve always a square matrix of tiles and hence a table.

Thus we have proven the following

**THEOREM 1.** The CTP is undecidable for regular sets.

To show the undecidability of the infinite CTP for regular sets, let

\[
\omega = \{ x/x \in L \cup M \lor (x \in A \land (\text{bottom}(x) = qa \Rightarrow \\
\Rightarrow q \neq \text{stop}(T))) \}.
\]

Obviously, defining \(L_2 = \Sigma_{\alpha\gamma\omega}\) for this \(\omega\), there are complete \((L_1, L_2)\)-tables of every size, if \(T\) doesn’t stop when started on the empty tape. On the other hand, if \(T\) stops within \(n\)-steps, there is no complete \((L_1, L_2)\)-table of size greater than \(n\). Thus we get, using again Lemma 1, the following
THEOREM 2. The infinite CTP is undecidable for regular sets.

We come to the last step of proof, namely to encode complete tables over arbitrary alphabets to complete tables over the two letter alphabet \{a, b\}. The encoding will be done in a way that single letters are represented as language tables. First we need some preparation. Consider an alphabet $X$ and the words $w = x_1 \ldots x_n \in X^*$ with $|w| = n$. We apply to these words the shift operations

$$\sigma_n^0(x_1 \ldots x_n) = x_1 \ldots x_n$$

and

$$\sigma_n^k(x_1 \ldots x_n) = x_{n-k+1} \ldots x_n x_1 \ldots x_{n-k} (0 < k < n).$$

Define for every $w$, the table $\Omega_w$ of size $|w|$ by

$$\Omega_w = \begin{bmatrix}
\sigma_n^0(w) \\
\vdots \\
\sigma_n^{n-1}(w)
\end{bmatrix}.$$

Now let $X = \{a_1, \ldots, a_m\}$. Let $p = p_1 \ldots p_m$ where $p_1, \ldots, p_m$ are the first $m$ primes and let $q_i = \frac{p_i}{p_1}$ for every $1 \leq i \leq m$. Assign to every $a_i$ the word

$$\omega_i = (ab^{p_i-1}q_i) (1 \leq i \leq m)$$

and the table $\Omega_i = \Omega_{\omega_i}$ of size $p$.

We prove the following useful results on $\Omega_i$:
LEMMA 2.

(1) \( \forall 1 \leq i \leq m \ \forall 1 \leq r, s, t \leq p: \)
\[
(\Omega_i[r,t] = \Omega_i[s,t] = a \rightarrow (R\Omega_i)_r = (R\Omega_i)_s).
\]

(2) \( \forall 1 \leq i, j \leq m, i \neq j \ \forall 1 \leq r, s \leq p \ \exists 1 \leq t \leq p: \)
\[
\Omega_i[r,t] = \Omega_j[s,t] = a.
\]

Proof. (1) is obviously true. To prove (2), consider \((R\Omega_i)_r\) and \((R\Omega_j)_s\). By definition
\[
(R\Omega_i)_r = \sigma^k_p((ab_i^{-1})^q_i)
\]
and
\[
(R\Omega_j)_s = \sigma^l_p((ab_j^{-1})^q_j)
\]
for some \(0 < k, l < p\). Choose \(k\) and \(l\) minimal, then
\(k < p_i\) and \(l < p_j\). By this
\[
(R\Omega_i)_r = \sigma^k_p((ab_i^{-1})^q_i) \cdot ab_i^{-1} p_i^{-k-1}
\]
and
\[
(R\Omega_j)_s = \sigma^l_p((ab_j^{-1})^q_j) \cdot ab_j^{-1} p_j^{-l-1}
\]
Consider the places where an \(a\) occurs. These are given by the formulas
\[
v = k + xp_i + 1 (0 \leq x < q_i)
\]
and
\[
v' = l + yp_j + 1 (0 \leq y < q_j).
\]
Hence we have to study the diophantine equation
\[
k + xp_i + 1 = l + yp_j + 1.
\]
It is easily checked, because \( \gcd(p_i, p_j) = 1 \), that there is always a solution with \( 0 \leq x < q_i \) and \( 0 \leq y < q_j \), which proves the lemma.

**LEMMA 3.** Every complete table \( A \) of \( \{\sigma^k_p(\omega_i) / 1 \leq i \leq m, 0 \leq k \leq p\} \) is a complete table of \( \{\sigma^k_p(\omega_j) / 0 \leq k < p\} \) for some \( j \); if \((RA)_j, (CA)_j \notin \{a,b\}^*aa\{a,b\}^*\) for \( 1 \leq l \leq p \).

**Proof.** Consider \( l \) with \( 1 \leq l \leq p \). Then \((RA)_l = \sigma^k_p(\omega_j) = (\Omega_j)^{k+1} \) for some \( j \) and \( k \) and \((RA)_{l+1} = \sigma^k_p(\omega_i) = (\Omega_i)^{k+1} \) for some \( i \) and \( k' \). By Lemma 2, \( i = j \). Otherwise \((CA)_t \notin \{a,b\}^*aa\{a,b\}^*\) for some \( t \). Obviously the same holds for the columns. Thus there exist \( j, i \) such that

\[
(RA)_l \notin \{\sigma^k_p(\omega_j) / 0 \leq k < p\}
\]

and

\[
(CA)_l \notin \{\sigma^k_p(\omega_i) / 0 \leq k < p\} \text{ for } 1 \leq l \leq p.
\]

Therefore

(i) \( \forall 1 \leq r \leq p \ \exists ! 1 \leq s \leq p_j : A[r,s] = a \)

and

(ii) \( \forall 1 \leq s \leq p \ \exists ! 1 \leq r \leq p_i : A[r,s] = a. \)

Let \( n = \#(\{(r,s) / A[r,s] = a \text{ and } 1 \leq r \leq p_i, 1 \leq s \leq p_j\}) \).

By (i), \( n = p_i \) and by (ii), \( n = p_j \). In conclusion \( i = j \), completing the proof.
Consider a language \( L \subseteq \{a_1, \ldots, a_m\}^* \). Associate to \( L \) the language

\[
L' = \bigcup_{k=1}^{p-1} \{ \sigma_p^k(\omega_i) / a_{i_1} \ldots a_{i_n} \in L \text{ and } 1 \leq i_\lambda \leq m \text{ for } 1 \leq \lambda \leq n \}.
\]

Obviously, every complete \( L \)-table of size \( n \) transfers to a complete \( L' \)-table of size \( n \cdot p \).

Now we show the converse correspondence, namely, that every complete \( L' \)-table transfers to a complete \( L \)-table. Consider a complete table \( A' \) of \( L' \). Then we can decompose \( A' \) into subtables \( B_{rs} \) of size \( p \).

\[
A' = \begin{bmatrix}
B_{11} & \cdots & B_{1n} \\
\vdots & & \vdots \\
B_{n1} & \cdots & B_{nn}
\end{bmatrix}.
\]

Consider an arbitrary \( B_{rs} = B \). By definition of \( L' \),

\[
(RB)_{\lambda}, (CB)_{\lambda} \in \{ \sigma_p^k(\omega_i) / 1 \leq i \leq m, 0 \leq k < p \} \setminus \{a, b\}^* aa\{a, b\}^*
\]

for \( 1 \leq \lambda \leq p \).

Lemma 3 yields \( (RB)_{\lambda}, (CB)_{\lambda} \in \{ \sigma_p^k(\omega_j(r, s)) / 0 \leq k < p \} \) for some fixed \( j(r, s) \). Hence for every row \( R \) of \( B_{r1} \ldots B_{rn} \)

\[
R \in \{ \sigma_p^k(\omega_j(r, 1)) \ldots \sigma_p^k(\omega_j(r, n)) / 0 \leq k < p \}
\]

and \( a_j(r, 1) \ldots a_j(r, n) \in L \) by definition of \( L' \).

Analogously, every column of the same "block" determines the same word in \( L \). Thus
\[ A[r, s] = a_j(r, s) \quad (1 \leq r, s \leq n) \]

is a complete \((n,n)\)-table of \(L\). In summary we have proven

**THEOREM 3.** The CTP and the infinite CTP are undecidable for regular sets \(L \subseteq \{a, b\}^* \).

*Remark:* It is quite easy to derive the result, that the \(n\)-CTP is NP-complete for regular sets. Obviously, the problem to decide for an arbitrary non-deterministic Turing machine \(T\) and a natural number \(n\) whether or not \(T\) accepts the empty word within \(n\) steps in NP-complete. Our construction exhibits a polynomial time reduction from this problem to the \(n\)-CTP for regular sets.

**REFERENCES**


*(Received March 6, 1986)*

Ulrike Brandt
Herman K.-G. Walter
Institut für Theoretische Informatik
Fachbereich Informatik
Technische Hochschule Darmstadt
Alexanderstrasse 24, D-6100 Darmstadt/Germany