SOME ALTERNATIVE WAYS TO FIND M-AMBIGUOUS BINARY WORDS CORRESPONDING TO A PARIKH MATRIX

Amrita Bhattacharjee[1] and Bipul Syam Purkayastha[2]

[1,2] Department of Computer Science, Assam University Silchar 788011, India

ABSTRACT

Parikh matrix of a word gives numerical information of the word in terms of its subwords. In this Paper an algorithm for finding Parikh matrix of a binary word is introduced. With the help of this algorithm Parikh matrix of a binary word, however large it may be can be found out. M-ambiguous words are the problem of Parikh matrix. In this paper an algorithm is shown to find the M- ambiguous words of a binary ordered word instantly. We have introduced a system to represent binary words in a two dimensional field. We see that there are some relations among the representations of M-ambiguous words in the two dimensional field. We have also introduced a set of equations which will help us to calculate the M- ambiguous words.

KEYWORDS

Parikh matrix, Subword, M- ambiguous words, word line

1. INTRODUCTION

In 1966 R.J.Parikh introduced a notion called Parikh Mapping [1]. This notion is an important tool in the theory of formal languages. With the help of this tool properties of words can be expressed numerically. In 2001 Mateescu et al [2] introduced the notion of Parikh matrix. The Parikh matrix gives more information about a word than the Parikh vector. A word is a finite or infinite sequence of symbols taken from a finite set called alphabet. This alphabet is an ordered alphabet. With every word over an ordered alphabet, a Parikh Matrix can be associated and it is a triangular matrix. All the entries of the main diagonal of this matrix is 1 and every entry below the main diagonal has the value 0 but the entries above the main diagonal provide information on the number of certain sub-words in w. An interesting aspect of the Parikh Matrix is that it has the classical Parikh vector as the second diagonal above the main diagonal. As such Parikh Matrix of a word gives information about Parikh vector of the same. Parikh Matrix still faces some challenging problems, such as it is not injective. Two words may have the same corresponding Parikh Matrix. But two words with the same Parikh vector have in many cases different Parikh matrices and thus the Parikh matrix gives more information about a word than the Parikh vector does. To overcome all the shortcomings, Parikh Matrix has become a research interest in related fields in recent years. In recent decades many techniques have been developed to solve complex problems of words using Parikh Matrix. We cite a few examples [3, 4, 5 …17] which has used subword occurrences and Parikh matrix for solving the problems of word.

Binary words are words made by $\{a, b\}$. There are many methods by which we can form the Parikh matrix of binary words, for example simple matrix product and use of various computational tools etc. Here we are introducing an algorithm for finding Parikh matrix of a binary word. With the help of this algorithm Parikh matrix of a binary word however large it may be can be found out. So, we are having various methods to find out the Parikh matrix of a binary word. But if we need to find the word corresponding to a Parikh matrix then there are few such readymade methods. To overcome this difficulty we introduce here an algorithm. This algorithm helps us to find the binary word corresponding to a Parikh matrix. Again we know that Parikh matrix is not injective i.e. corresponding to a Parikh matrix there may be more than one word, this property is known as M-ambiguity. The words corresponding to a single Parikh matrix are known as amiable words. With this algorithm all the amiable words can be found at the same time. Various methods are used for finding M-ambiguous binary words. The algorithm we have introduced in this paper is also an effort in this regard. With the help of this algorithm all the M-ambiguous words corresponding to a 3x3 Parikh matrix can be found instantly. One just has to enter a 3x3 Parikh matrix. If the matrix is not a Parikh matrix then there will be simply no corresponding word. So this algorithm will prove a useful tool in the research area of M-ambiguity. In this paper we introduce a notion regarding the representation of binary words in two dimensional fields. This notion gives an interesting point of view regarding M- ambiguity. We have also introduced a set of equations which give us the corresponding binary word from a given 3x3 Parikh matrix.

The paper is organized as follows. The following section 2 reviews the basic preliminaries of Parikh Matrix. In section 3 related works are discussed. Section 4 goes toward developing the algorithm for display Parikh Matrix of a sequence over binary alphabet. Section 5 gives an algorithm which gives binary words (may be one word or more than one word due to M-ambiguity) corresponding to a Parikh matrix; In Section 6, notion of representation of words in two dimension is given; in section 7 result analysis are presented. Section 8 gives proposed equations to find out the binary sequences corresponding to a given Parikh Matrix. We conclude the paper in Section 9 by summarizing the observations.

2. PRELIMINARIES

Throughout this paper N will denote the set of natural numbers including 0. First we recall some definitions.

Ordered alphabet: An ordered alphabet is a set of symbols $\Sigma = \{a_1, a_2, a_3, \cdots, a_n\}$ where the symbols are arranged maintaining a relation of order ("$<$") on it. For example if $a_1 < a_2 < a_3 < \cdots < a_n$, then we use notation: $\Sigma = \{a_1, a_2, a_3, \cdots, a_n\}$

Word: A word is a finite or infinite sequence of symbols taken from a finite set called an alphabet. Let $\Sigma = \{a_1, a_2, a_3, \cdots, a_n\}$ be the alphabet. The set of all words over Σ is Σ^*. The empty word is denoted by λ.

$|w|_{a_i}$: Let $a_i \in \Sigma = \{a_1, a_2, a_3, \cdots, a_n\}$ be a letter. The number of occurrences of a_i in a word $w \in \Sigma^*$ is denoted by $|w|_{a_i}$.

Sub -word: A word u is a sub- word of a word w, if there exists words $x_1 \cdots x_n$ and $y_0 \cdots y_n$, (some of them possibly empty), such that $u = x_1 \cdots x_n$ and $w = y_0 x_1 y_1 \cdots x_n y_n$. For example if $w = abaabcac$ is a word over the alphabet $\Sigma = \{a, b, c\}$ then $baca$ is a sub-word of w. Two occurrences of a sub-word are considered different if they differed by at least one position of some letter. In the word $w = abaabcac$, the number of occurrences of the word $baca$ as a sub-word of w is $|w|_{baca} = 2$.

Parikh vector: The Parikh vector is a mapping $\Psi : \Sigma^* \to N \times N$ where $\Sigma = \{a_1, a_2, a_3, \cdots, a_n\}$ and N is the set of natural numbers including 0, such that for a word w in Σ^*, $\Psi(w) = (|w|_{a_1}, |w|_{a_2}, |w|_{a_3}, \cdots, |w|_{a_n})$ with $|w|_{a_i}$ denoting the number of occurrences of the letter $a_i \in w$. For example, for the word $w = abaabcac$ the Parikh vector is $(4, 2, 2)$.

Triangle matrix: A triangle matrix is a square matrix $m = (m_{ij})_{1 \le i, j \le n}$ such that:

1. $m_{ij} \in N \quad (1 \le i, j \le n)$,
2. $m_{ij} = 0$ for all $1 \le j < i \le n$,
3. $m_{ii} = 1 \quad (1 \le i \le n)$.

Parikh matrix: Let $\Sigma = \{a_1 < a_2 < a_3 < \cdots < a_n\}$ be an ordered alphabet, where $n \ge 1$. The Parikh matrix mapping, denoted Ψ_{M_n}, is the homomorphism $\Psi_{M_n} : \Sigma^* \to M_{n+1}$ defined as follows:

if $\Psi_{M_n}(a_q) = (m_{ij})_{1 \le i, j \le n+1}$ then $m_{i,i} = 1$, $m_{q,q+1} = 1$ and all other elements are zero.

M-ambiguous or Amiable words: Two words $\alpha, \beta \in \Sigma^*$ $(\alpha \ne \beta)$ over the same alphabet Σ may have the same Parikh matrix. Then the words are called amiable or M-ambiguous.

The words $baaabaa$ and $ababaaa$ has the same Parikh Matrix $\begin{pmatrix} 1 & 5 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix}$. So these two words are amiable.

M-unambiguous words: A word w is said to be M-unambiguous if there is no word w' for which $\Psi_{M_n}(w) = \Psi_{M_n}(w')$.

3. RELATED WORKS

Since the introduction of the notion of Parikh vector in 1966 [1] continuous research works are going on in this field. In this introductory paper [1] certain properties of context-free or type 2 grammars are investigated. In particular, questions regarding structure, possible ambiguity and relationship to finite automata are considered. Some important results are also presented. A sharpening of Parikh mapping namely Parikh matrix is introduced in [2] and this matrix representation gives more information than Parikh vector does. With the extension of Parikh matrix an interesting interconnection between mirrors images of words and inverses of matrices are investigated in [3]. Researchers [4] have presented ratio property of two words. This property is a sufficient condition for the words uv & vu to have the same Parikh matrix. In this paper [5] universal languages for Parikh matrices is introduced and studied. In [6] M-unambiguity is studied both in terms of words and matrices and several criteria for M-unambiguity are provided in both cases. In this paper [7] palindromic amicable words are studied in the context of binary words. Researchers [8] have introduced subword condition. Various characterization and decidability results for languages subword conditions are discussed. Quasi-uniform event from the earlier automata theory plays a central role in the investigation of characterisation. In this paper [9] Parikh Matrices over tertiary alphabet are investigated. Algorithm is developed to display Parikh Matrices of words over tertiary alphabet. A set of equations for finding tertiary words from the respective Parikh matrix is introduced. In this paper [10] the notion of a subword history closely related to Parikh matrices is introduced and obtained a sequence of general results. A general inequality of Cauchy type for subword occurrences is established. In [11] a natural extension of Parikh matrix and a set of properties for this kind of matrices are investigated. The combination of Parikh matrix and this extension give a more powerful tool for the study of algebraic properties of words. Different characterizations of pairs of words having the same Parikh matrix are investigated in this paper [12]. In this paper [13] certain inequalities, as well as information about subword occurrences sufficient to determine the whole word uniquely are studied. Some algebraic considerations, facts about forbidden subwords, as well as some open problems are also included. Researchers [14] have investigated the numerical quantity $|w|_u$, the number of occurrences of a word *u* as a (scattered) subword of a word *w*. Parikh matrices recently introduced have these quantities as their entries. According to the main result in this paper, no entry in a Parikh matrix, no matter how high the dimension can be computed in terms of the other entries. In [15] Amiable words are investigated. It is shown that all the words having the same Parikh matrix can be obtained one from another by applying only two types of transformations. It is also shown that mirrors of two amiable words are also amiable. The paper [16] investigates some properties of the set of binary words having the same Parikh matrix. These words are named as amiable words. Investigations are done on the equivalence class. A characterisation theorem concerning a graph associated to an equivalence class of amiable words, and some basic properties of a rank distance is discussed. In this paper [17] ratio property of words are investigated. Concept of ratio property and weak ratio property are extended for nth order alphabet. A relationship of ratio property with M- ambiguity is established. Various lemmas already proved about ratio property over ternary alphabet are investigated for tertiary alphabets. M-ambiguous words are formed by concatenating words satisfying ratio property.

4. ALGORITHM TO DISPLAY PARIKH MATRIX CORRESPONDING TO A WORD

There are many methods by which we can form the Parikh matrix of binary words, for example simple matrix product and use of various tools of computing matrix product etc. Here an algorithm for finding Parikh matrix of a binary word is introduced. With the help of this algorithm Parikh matrix of a binary word, however large it may be can be found out.

4.1 Algorithm: The following pseudo code gives instantly the Parikh matrix of a binary sequence.

```
01  Initialise a word = ‘ w’
02  Set len = length of w
03  For i = 0 to len do
04    Calculate total number of a, ab in w.
05    Calculate total number of b in w.
```

```
End.
    // create a matrix (a_ij) of order M (=3)
For i = 0 to M do
    For j = 0 to M do
        If (i = j)
            a_ij = 1
        else If (i > j)
            a_ij = 0
        else
            If (i = 0 & j = 1)
                a_ij = total number of 'a'
            If (i = 0 & j = 2)
                a_ij = total number of 'ab'
            If (i = 1 & j = 2)
                a_ij = total number of 'b'
    End
End
```

4.2 Application of above algorithm:

Example1. The binary word $\xi_1 = abab\underbrace{a\cdots a}_{10}\underbrace{b\cdots b}_{15}$ has the Parikh matrix

$$\Psi_{M_2}(\xi_1) = \begin{pmatrix} 1 & 12 & 183 \\ 0 & 1 & 17 \\ 0 & 0 & 1 \end{pmatrix}$$

Example2. The binary word $\xi_2 = abababab\underbrace{a\cdots a}_{30}\underbrace{b\cdots b}_{29}$ has the Parikh matrix

$$\Psi_{M_2}(\xi_2) = \begin{pmatrix} 1 & 34 & 996 \\ 0 & 1 & 33 \\ 0 & 0 & 1 \end{pmatrix}$$

Example3. The binary word $\xi_3 = \underbrace{b\cdots b}_{20}ababababababab\underbrace{a\cdots a}_{38}$ has the Parikh matrix

$$\Psi_{M_2}(\xi_3) = \begin{pmatrix} 1 & 45 & 28 \\ 0 & 1 & 27 \\ 0 & 0 & 1 \end{pmatrix}.$$

5. ALGORITHM TO DISPLAY WORDS CORRESPONDING TO A PARIKH MATRIX

Various methods are used for finding M-ambiguous words for binary words; this algorithm is also an effort in this regard. With the help of this algorithm all the M-ambiguous words corresponding to a 3x3 Parikh matrix can be found instantly. One just has to enter a 3x3 Parikh matrix. If the matrix is not a Parikh matrix then there will be simply no corresponding word.

5.1 Algorithm: The following pseudo code gives instantly the ternary sequences corresponding to a Parikh matrix.

Input a matrix $A_{i,j}$ of order 3×3
$l = (A_{0,1} + A_{1,2})$
Store $'l'$ time of $'ab'$ in w and consider it as an initial word $A_{i,j}$
Store w into L (a list)
len = Number of elements of (L)
For $i = 1$ to len do
    W = word at i th position of L
        For $j = 1$ to length of W do
            Store W by removing character at j^{th} position
        End
    Remove all duplicate copies of words from L
    Remove W from L if numbers of a in $W \neq A_{0,1}$ & numbers of b in $W \neq A_{1,2}$
If it is removed then update i by $i-1$
update len = Number of elements of (L)
if all the words of L is of length l
  End
  For $k = 1$ to len do
      Perform Algorithm 4.1 with each word at k^{th} position
      It gives matrix X_{ij} of size 3×3
      Check each of X_{ij} with A_{ij}
      If it matches then display ‘YES’ otherwise display ‘NO’.
  End.

5.2 Application of above algorithm

a) Let $\Sigma=\{a<b\}$ and the Parikh matrix be $\Psi_{M_2}(\zeta_1)=\begin{pmatrix}1&3&5\\0&1&3\\0&0&1\end{pmatrix}$, the set of amiable words having this Parikh matrix is $C=\{ababba, abbaab, baabab\}$

b) Let $\Sigma=\{a<b\}$ and the Parikh matrix be $\Psi_{M_2}(\zeta_2)=\begin{pmatrix}1&4&7\\0&1&3\\0&0&1\end{pmatrix}$, the set of amiable words having this Parikh matrix is $C=\{ababaab, baaabab, aabbaba, abaabba\}$

c) Let $\Sigma=\{a<b\}$ and the Parikh matrix be $\Psi_{M_2}(\zeta_3)=\begin{pmatrix}1&7&9\\0&1&3\\0&0&1\end{pmatrix}$, the set of amiable words having this Parikh matrix is

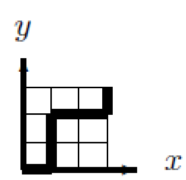
$$C=\{aabababaaa, abaabaabaa, ababaaaaba, baaaababaa, baaabaaaba, baabaaaaab, aabbaaabaa, abaaabbaaa, abbaaaaaab, aaabbbaaaa\}$$

d) Let $\Sigma=\{a<b\}$ and the Parikh matrix be $\Psi_{M_2}(\zeta_4)=\begin{pmatrix}1&5&9\\0&1&5\\0&0&1\end{pmatrix}$, the set of amiable words having this Parikh matrix is

$$C=\{bbaabababa, bbabaabaab, babbaababa, bababbaaba, bababbaaba, babababbaa, abbbabaaba, abbabbabaa, bbbaaaabab, bbaabbaaab, bbabaaabba, baabbbabaa, abbbbaaaab, abbbaabbaa, ababbbbaaa, bbaaabbbaa\}$$

6. Introduction of the Notion of Two Dimensional Representation of Binary Words

In this section a method to represent binary words in a two dimensional area is introduced. For this one first have to draw two perpendicular axes X and Y (say) intersecting at a point. Considering the coordinate of the intersecting point as (0, 0), in the X-axis we take a 's and in the Y-axis we take b 's. Now to draw the graph of the word, this graph is named as word line. We start from $(0,0)$ i.e. the intersection point of the two axes and go on describing the word w as follows: If the first letter of the word is a then we move to $(1,0)$, if the second letter is again a then we move to $(2,0)$ and if the second letter is b then we move to $(1,1)$. But if the first letter of the word is b then we move to $(0,1)$, and thus go on describing the path of the word in the two dimensional graph. Let $M_\alpha = \begin{pmatrix} 1 & |w|_a & |w|_{ab} \\ 0 & 1 & |w|_b \\ 0 & 0 & 1 \end{pmatrix}$ be the Parikh matrix corresponding to the word w. Now we draw two line segments. One line segment is parallel to the Y-axis through the point $\left(|w|_a, 0\right)$ another line segment is parallel to the X-axis through the point $\left(0, |w|_b\right)$. Thus we shall get a closed bounded area as shown in figure 1: this area is either a square or a rectangle depending upon $|w|_a = |w|_b$ or $|w|_a \neq |w|_b$ .Now we divide the area into $|w|_a + |w|_b$ lines. We draw $|w|_a$ equidistant lines parallel to the Y-axis and $|w|_b$ equidistant lines parallel to the X-axis. These lines divide the prescribed area into $|w|_a \times |w|_b$ squares. The line traced by the word divides the rectangle in two parts. The upper part is bounded by the lines Y-axis, the line parallel to the X-axis through the point $\left(0, |w|_b\right)$, and the line traced by the word itself. The lower part is bounded by the lines X-axis, the line parallel to the Y-axis through the point $\left(|w|_a, 0\right)$, and the line traced by the word itself. We name the line traced by the word as Word line. The numbers of squares on the upper part is exactly equal to $|w|_{ab}$ and the number of squares on the lower part helps us to add some property in the field of amiable words. We can see that all the amiable words have the same area covered. Conversely the words corresponding to the same area covered are M-ambiguous words. So we draw a word in the two dimensional field in the same way defined above and with the same $|w|_a$ and $|w|_b$ if we can draw another word which have the same area covered then the two words will be amiable. For example let us draw the word $abbaab$ on two dimensional areas.

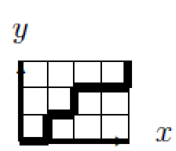


$abbaab$

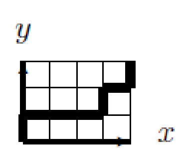
7. RESULT ANALYSIS

I. The two dimensional representation of the matrix $M_{\alpha_1} = \begin{pmatrix} 1 & 4 & 7 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{pmatrix}$

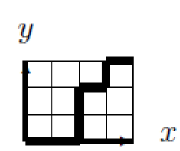
is given as follows:



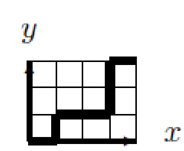
1.$ababaab$



2.$baaabab$



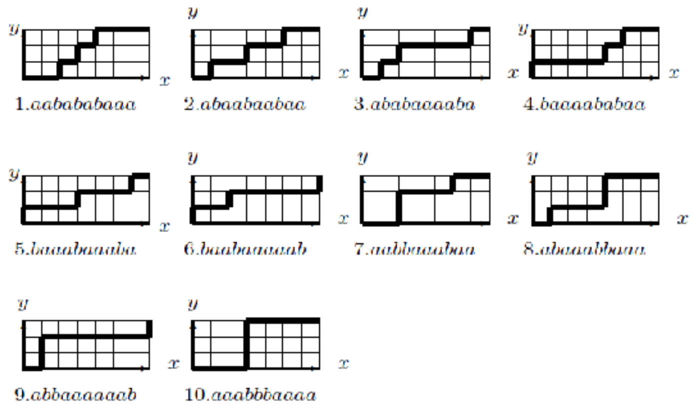
3.$aabbaba$



4.$abaabba$

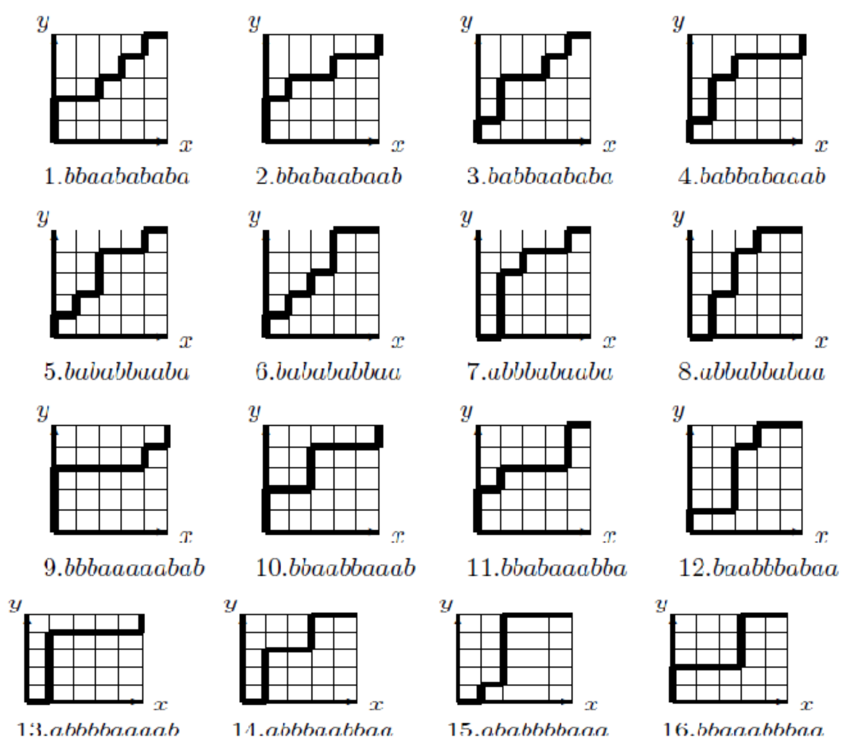
II. The two dimensional representation of the matrix $M_{\alpha_2} = \begin{pmatrix} 1 & 7 & 9 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{pmatrix}$

is given as follows:



III. The two dimensional representation of the matrix $M_{\alpha_3} = \begin{pmatrix} 1 & 5 & 9 \\ 0 & 1 & 5 \\ 0 & 0 & 1 \end{pmatrix}$

is given as follows:



8. SET OF EQUATIONS FOR PARIKH MATRIX TO WORDS:

We now introduce a new set of equations corresponding to a binary word $\beta \in \Sigma^*$. Let $\Sigma = \{a < b\}$ be a binary ordered alphabet and $\beta \in \Sigma^*$ be a binary sequence.

If $|w|_a = f$, $|w|_b = g$ then β can be represented in the following form: $\beta = a^{x_1} b^{y_1} a^{x_2} b^{y_2} \cdots a^{x_{f+g}} b^{y_{f+g}}$,the Parikh matrix

$\Psi_{M_2}(\beta) = \begin{pmatrix} 1 & f & h \\ 0 & 1 & g \\ 0 & 0 & 1 \end{pmatrix}$ corresponds to this word if and only if x_i = either 0 or 1and y_j = either 0 or 1 is a solution of the following system of equations:

$$\sum_{i=1}^{f+g} x_i = f \tag{a}$$

$$\sum_{j=1}^{f+g} y_j = g \tag{b}$$

$$\sum_{i=1}^{f+g} x_i \sum_{j=i}^{f+g} y_j = h \tag{c}$$

For clear understanding we take the example of the following Parikh matrix.
Example 1: Let

$$\Psi_{M_2}(\varsigma_1) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

be a Parikh matrix. Here $|w|_a = 1$, $|w|_b = 1$. Then $\varsigma_1 \in \Sigma^*$ is a binary sequence corresponds to the above matrix. Here $f+g=1+1=2$. So ς_1 can be represented in the following form: $\varsigma_1 = a^{x_1} b^{y_1} a^{x_2} b^{y_2}$

The Parikh Matrix corresponds to this word if and only if x_i = either 0 or 1, y_j = either 0 or 1, is a solution of the following system of equations:

$$\sum_{i=1}^{2} x_i = 1 \tag{1}$$

$$\sum_{j=1}^{2} y_j = 1 \tag{2}$$

$$\sum_{i=1}^{2} x_i \sum_{j=i}^{2} y_j = 0 \tag{3}$$

Now from (1), we get, $x_1 + x_2 = 1$ from (2), we get, $y_1 + y_2 = 1$ from (3), we get.
$x_1(y_1 + y_2) + x_2 y_2 = 0 \Rightarrow x_1(1) + x_2 y_2 = 0$ [from (2)] $\Rightarrow y_2 = 0$ [using (1)]

$\therefore y_1 = 1$ again from (3) we have

$$x_1(y_1 + y_2) + x_2 y_2 = 0 \Rightarrow x_1(1+0) + x_2 \cdot 0 = 0 \Rightarrow x_1 = 0$$

So the word $\varsigma_1 = a^{x_1} b^{y_1} a^{x_2} b^{y_2}$ is $\varsigma_1 = a^0 b^1 a^1 b^0 = ba$

Example 2: Let

$$\Psi_{M_2}(\varsigma_2) = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

be a Parikh matrix. Here $|w|_a = 1$, $|w|_b = 1$. Then $\varsigma_2 \in \Sigma^*$ is a binary sequence corresponds to the above matrix. Here $f + g = 1 + 1 = 2$. So ς_2 can be represented in the following form: $\varsigma_2 = a^{x_1} b^{y_1} a^{x_2} b^{y_2}$

The Parikh Matrix corresponds to this word if and only if x_i = either 0 or 1, y_j = either 0 or 1, is a solution of the following system of equations:

$$\sum_{i=1}^{2} x_i = 1 \tag{1}$$

$$\sum_{j=1}^{2} y_j = 1 \tag{2}$$

$$\sum_{i=1}^{2} x_i \sum_{j=i}^{2} y_j = 1 \tag{3}$$

Now from (1), we get, $x_1 + x_2 = 1$ from (2), we get, $y_1 + y_2 = 1$ from (3), we get.

$x_1(y_1 + y_2) + x_2 y_2 = 1 \Rightarrow x_1(1) + x_2 y_2 = 1$ [from (2)]

$\Rightarrow$ either $x_1 = 1,\ x_2 = 0,\ y_2 = 0$ or $x_1 = 0, x_2 = 1, y_2 = 1$

if $y_2 = 0$ then from (2) we have $y_1 = 1 \Rightarrow$ either $x_1 = 1, y_1 = 1$ or $x_2 = 1, y_2 = 1$

So the word $\varsigma_2 = a^{x_1} b^{y_1} a^{x_2} b^{y_2}$ is either $\varsigma_2 = a^1 b^1 a^0 b^0 = ab$ or $\varsigma_2 = a^0 b^0 a^1 b^1 = ab$.

This is how we can use the proposed set of equations to find the corresponding word from a 3 x 3 Parikh matrix.

9. CONCLUSIONS

In this paper we have given one algorithm for finding the Parikh matrix corresponding to a word. This helps us to find Parikh matrix of a binary word big or small instantly. M-ambiguous words are the problem of Parikh matrices. In this paper an algorithm is shown to find the M- ambiguous words of a binary ordered word. A system to represent binary words in a two dimensional field is introduced. There are some relations among the representations of M-ambiguous words in the two dimensional field. The area covered by line depicted by the word and the X-axis and the line parallel to the Y-axis though the point $\left(0,|w|_b\right)$ is the same for all M-ambiguous words corresponding to a 3x3 Parikh matrix. The number of squares covered by the word line, Y-axis and the line parallel to the X-axis though the point is the same as the number $|w|_{ab}$. It is seen that if one word corresponding to a Parikh matrix is known then those words which have the same area covered below the word line and with the same $|w|_a + |w|_b$ are all M- ambiguous words. The reason behind this property is yet to be investigated. a set of equations by which also we can calculate out the M-ambiguous words of a Parikh matrix is also introduced. One can find M-ambiguous words by using this set of equations.

REFERENCES

[1] R.J.Parikh, (1966) "On the context-free languages" Journal of the Association for Computing Machinery, Vol No.13 pp 570-581.

[2] A. Mateescu, A. Salomaa, K. Salomaa, S.Yu, (2001) "A sharpening of the Parikh Mapping" Theoret. Informetics Appl., Vol No. 35 pp 551-564.

[3] A. Mateescu, A. Salomaa, K. Salomaa, S.Yu: On an extension of the Parikh mapping, T.U.C.S Technical Report No 364.

[4] K. G. Subramanian, A. M. Huey, A. K. Nagar, (2009) "On Parikh matrices" Int. J. Found. Comput. Sci. Vol.20 No.2 pp211-219.

[5] C. Ding, A. Salomaa, (2006) "On some problems of Mateescu concerning sub word occurences" Fundamenta Informaticae Vol No. 72 pp1-15.

[6] V.N. Serbˇanutˇa, (2006) "Injectivity of the Parikh matrix mappings revisited" Fundamenta Informaticae Vol No. XX pp1–19, IOS Press.

[7] A. Atanasiu, C. M.Vide, A. Mateescu, (2001) "On the injectivity of the Parikh matrix mapping" Fundam. Informa. Vol No.46 pp 1-11.

[8] A. Salomaa et al, (2006) "Subword conditions and subword histories" Information and Computation Vol No.204 pp1741–1755.

[9] Amrita Bhattacharjee, Bipul Syam Purkayastha, (2014) "Parikh Matrices and Words over Tertiary Ordered Alphabet" Int. J. of Computer Applications, Vol 85 No.4 pp10–15.

[10] A. Mateescu, A. Salomaa, S. Yu, (2004) "Subword histories and Parikh matrices" J. Comput. Syst. Sci., Vol No.68 pp 1–21.

[11] T.-F. S¸ erbˇanut¸ˇa, (2004) "Extending Parikh matrices" Theoretical Computer Science, Vol No.310 pp 233–246.

[12] S. Fosse, G. Richmomme, (2004) "Some characterisations of Parikh matrix equivalent binary words" Information Processing Letters. Vol.92 No.2 pp77–82.

[13] A. Salomaa, (2005) "Connections between subwords and certain matrix mappings" Theoretical Computer Science, Vol No.340 pp 188–203.

[14] A. Salomaa, (2006) "Independence of certain quantities indicating subword occurences" Theoretical Computer Science. Vol.362 No.1 pp222–231.

[15] Adrian Atanasiu, Radu Atanasiu, Ion Petre, (2008) "Parikh matrices and amiable words" Theoretical Computer Science Vol No.390 pp 102–109.

[16] A. Atanasiu, (2007) "Binary amiable words" Int. J. Found. Comput. Sci Vol.18 No.2 pp387-400.

[17] Amrita Bhattacharjee, Bipul Syam Purkayastha, (2014) "Application of ratio property in searching of M-ambiguous words and its generalization" Proceedings of Third International Conference on Soft computing for Problem Solving ,SocProS 2013, AISC, Vol 259, (Selected for publication), Springer India.