A Note on Adaptive 2D-H Strings [1]

Ye-In Chang and Hsing-Yen Ann

Dept. of Applied Mathematics
National Sun Yat-Sen University
Kaohsiung, Taiwan
R.O.C.
{E-mail: changyi@math.nsysu.edu.tw}
{Tel: 887-7-5252000 (ext. 3819)}
{Fax: 886-7-5253809}

Abstract

A picture is defined to be *ambiguous* if there exists more than one different reconstructed picture from its representation. In this paper, we first give an ambiguous case based on the adaptive 2D-H string representation [1]. Next, we show how to avoid the ambiguous cases.

(*Keywords*: 2D strings, 2D-H strings, adaptive 2D-H strings, image databases, symbolic pictures)

[1] This research was supported in part by the National Science Council of Republic of China under Grant No. NSC-87-2213-E-110-014.

1 Introduction

In [3], Chang and Li has proposed 2D-H strings, which can be viewed as a combination of quadtrees [4] and 2D strings [2]. Using the 2D-H string, the hierarchical symbolic pictures can be represented efficiently in terms of space complexity. Although the 2D-H string data structure has been proven to be an efficient approach to represent and to manipulate symbolic pictures, in [1], Chang and Lin has discovered some redundancies existing in those data representations. Therefore, they proposed another alternative, called adaptive 2D-H strings, for representing the relationships among the objects in an image.

In [1], Chang and Lin has presented an algorithm for converting symbolic pictures of any size into adaptive 2D-H strings. They show that their adaptive 2D-H string can work well for many unbalanced non-square small pictures, which frequently exist in our real environment. However, based on Chang and Lin's procedure to construct the adaptive 2D-H string, *ambiguous* cases can occur, where a picture is defined to be *ambiguous* if there exists more than one different reconstructed picture from its representation. Therefore, in this paper, we first give an ambiguous case based on the adaptive 2D-H string representation [1]. Next, we show how to avoid the ambiguous cases.

2 An Ambiguous Case

Take Figure 1 as an example, where picture f_1 and f_2 are two different pictures while they contain the same 4 symbols occupying 12 cells. The corresponding decomposition steps for pictures f_1 and f_2 are shown in Figure 2 and Figure 3, respectively.

Moreover, the corresponding adaptive 2D-H string representation for pictures f_1 and f_2 are as follows:

$$\begin{array}{l}
\text{adaptive 2D-H}(f_1) \\
= b_N b_S s_N s_S \\
= 11\ s_N s_S \\
= 11\ \mathbf{10}\ s_{NN} s_{NS}\ \mathbf{11}\ s_{SN} s_{SS} \\
= 11\ \mathbf{10}\ \underline{1001AB}\ \mathbf{11}\ \underline{1000C}\ \underline{01D}
\end{array}$$

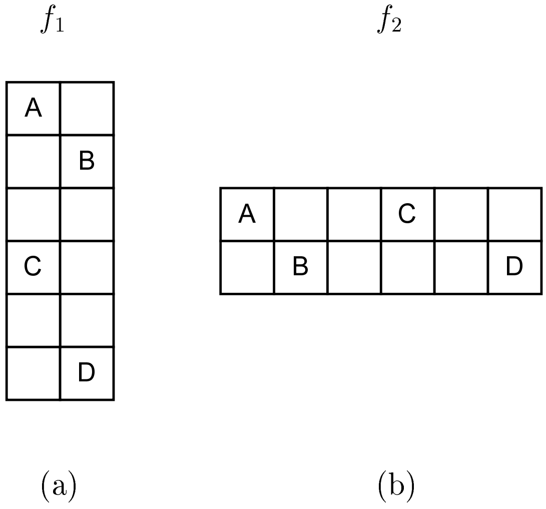
adaptive 2D-H(f_2)
$= b_W b_E s_W s_E$
$= 11\ s_W s_E$
$= 11\ \mathbf{10}\ s_{WW} s_{WE}\ \mathbf{11}\ s_{EW} s_{EE}$
$= 11\ \mathbf{10}\ \underline{1001AB}\ \mathbf{11}\ \underline{1000C}\ \underline{01D}$

3 The Revised Version of the Adaptive 2D-H Strings

From the above example, we show that pictures represented in the adaptive 2D-H strings can be ambiguous. In this example, pictures f_1 and f_2 have the same corresponding quadtree as shown in Figure 4. To overcome this problem, we provide an answer. We can avoid the ambiguous case by adding the size information of a picture, say $m_1 \times m_2$, at the end of the corresponding adaptive 2D-H string. The *Reconstruct* procedure presented in the Appendix shows how to reconstruct a picture based on the revised version of the adaptive 2D-H string without causing any ambiguous. In this *Reconstruct* procedure, we use the size information of a picture f, say $m \times n$, to guide us how to decompose the adaptive 2D-H string, just the same case as how the picture f is segmented. In this way, obviously, when $m_1 \neq m_2$ or $n_1 \neq n_2$, two pictures f_1 (with size $m_1 \times n_1$) and f_2 (with size $m_2 \times n_2$) will be distinguished well even they have the same adaptive 2D-H string representation.

4 Conclusion

The adaptive 2D-H string representation has been proposed to remove the redundancy existing in the 2D-H string representation. However, the concise representation of the adaptive 2D-H string can cause ambiguous cases. In this paper, we have shown such a case and have provided an answer to avoid the ambiguous case.



(a) (b)

Figure 1: An example

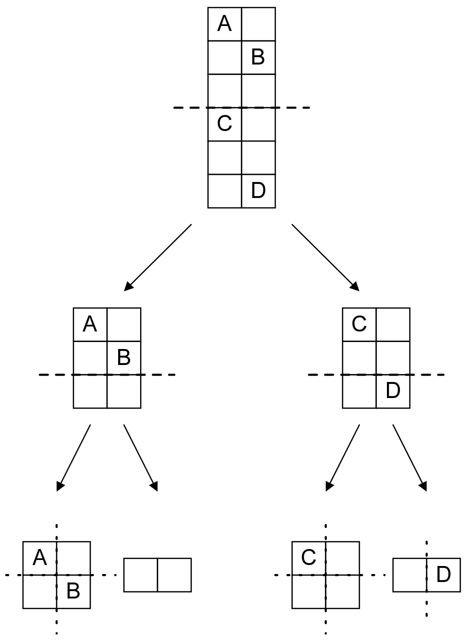


Figure 2: Decomposition steps for picture f_1

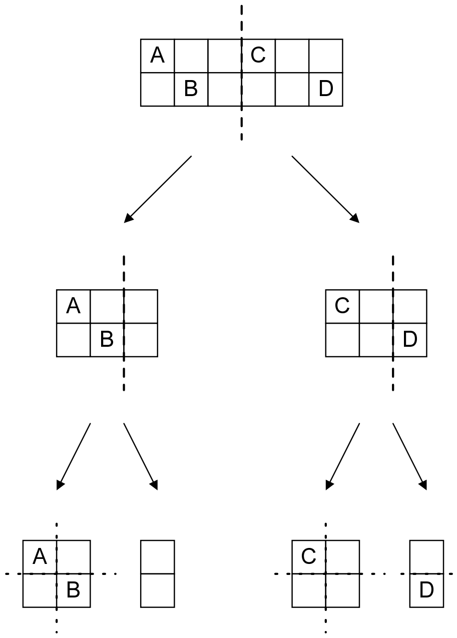


Figure 3: Decomposition steps for picture f_2

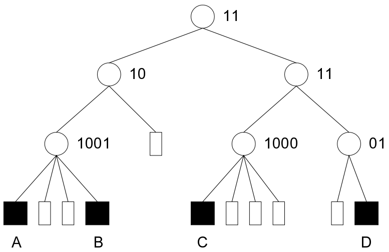


Figure 4: The quadtree of picture f_1 (f_2)

References

[1] Chang, C. C. and Lin, D. C. (1996). A Spatial Data Representation: An Adaptive 2D-H String. *Pattern Recognition Letters,* 17 (2), 175-185.

[2] Chang, S. K., Shi, Q. Y. and Yan. C. W. (1987). Iconic Indexing by 2-D Strings, *IEEE Trans. on Pattern Analysis and Machine Intelligence,* PAMI-9 (3), 413-428.

[3] Chang, S. K. and Li, Y. (1988). Representation of Multi-Resolution Symbolic and Binary Pictures Using 2D-H Strings, *Proc. IEEE Workshop on Languages for Automata,* Maryland, 190-195.

[4] Samet, H. (1984). The Quadtree and Related Data Structure, *ACM Comput. Surveys,* 16 (2), 187-260.

[5] Samet, H. (1990). *Applications of Spatial Data Structure,* Addison-Wesley, Reading, MA.

Appendix

Procedure Reconstruct(f,m,n)

Input: (1) the size of a symbolic picture f, m, n;
(2) a global variable S, the adaptive 2D-H string of f
Output: the symbolic picture f

```
IF (min($m$,$n$) $> 2$) THEN        % quadrant segmentation %
BEGIN
    set $f_1$, $f_2$, $f_3$ and $f_4$ to be NW, SW, NE and SE
    quadrants subpictures of $f$, respectively
    % let $S_i$ be the $i$th bit of $S$ from the left side %
    FOR $i = 1$ to 4
        $b_i := S_i$
    $S \leftarrow S$ shift left 4 bits
    IF ($b_1 = 1$) THEN
        Reconstruct($f_1$,$\lceil 1/2m \rceil$,$\lceil 1/2n \rceil$)        % NW %
    IF ($b_2 = 1$) THEN
        Reconstruct($f_2$,$\lfloor 1/2m \rfloor$,$\lceil 1/2n \rceil$)        % SW %
    IF ($b_3 = 1$) THEN
        Reconstruct($f_3$,$\lceil 1/2m \rceil$,$\lfloor 1/2n \rfloor$)        % NE %
    IF ($b_4 = 1$) THEN
        Reconstruct($f_4$,$\lfloor 1/2m \rfloor$,$\lfloor 1/2n \rfloor$)        % SE %
END
ELSE IF ($m \leq 2$ and $n > 2$) THEN        % column segmentation %
BEGIN
    set $f_1$ and $f_2$ to be W and E quadrant subpictures of $f$
    FOR $i = 1$ to 2
        $b_i := S_i$
    $S \leftarrow S$ shift left 2 bits
    IF ($b_1 = 1$) THEN
        Reconstruct($f_1$,$m$,$\lceil 1/2n \rceil$)        % W %
    IF ($b_2 = 1$) THEN
        Reconstruct($f_2$,$m$,$\lfloor 1/2n \rfloor$)        % E %
END
ELSE IF ($m > 2$ and $n \leq 2$) THEN        % row segmentation %
BEGIN
    set $f_1$ and $f_2$ to be N and S quadrant subpictures of $f$
    FOR $i = 1$ to 2
        $b_i := S_i$
    $S \leftarrow S$ shift left 2 bits
    IF ($b_1 = 1$) THEN
        Reconstruct($f_1$,$\lceil 1/2m \rceil$,$n$)        % N %
    IF ($b_2 = 1$) THEN
        Reconstruct($f_2$,$\lfloor 1/2m \rfloor$,$n$)        % S %
```

```
END
ELSE        % the elementary unit of decomposition %
BEGIN
    IF ($m = 2$ and $n = 2$) THEN        % type-1 unit %
    BEGIN
        set $f_1$, $f_2$, $f_3$ and $f_4$ to be NW, SW, NE and SE
        quadrants subpictures of $f$, respectively
        FOR $i = 1$ to 4
            $b_i := S_i$
        $S \leftarrow S$ shift left 4 bits
        FOR $i = 1$ to 4
            IF ($b_i = 1$) THEN
            BEGIN
                $B \leftarrow$ the first symbol from the left side of $S$
                output $B$ in $f_i$
                $S \leftarrow S$ shift left 1 symbol
            END
    END
    ELSE IF ($m = 2$) THEN        % type-2 unit %
    BEGIN
        set $f_1$ and $f_2$ to be N and S quadrant subpictures of $f$
        FOR $i = 1$ to 2
            $b_i := S_i$
        $S \leftarrow S$ shift left 2 bits
        FOR $i = 1$ to 2
            IF ($b_i = 1$) THEN
            BEGIN
                $B \leftarrow$ the first symbol from the left side of $S$
                output $B$ in $f_i$
                $S \leftarrow S$ shift left 1 symbol
            END
    IF ($n = 2$) THEN        % type-3 unit %
    BEGIN
        set $f_1$ and $f_2$ to be E and W quadrant subpictures of $f$
        FOR $i = 1$ to 2
            $b_i := S_i$
        $S \leftarrow S$ shift left 2 bits
        FOR $i = 1$ to 2
            IF ($b_i = 1$) THEN
            BEGIN
                $B \leftarrow$ the first symbol from the left side of $S$
                output $B$ in $f_i$
                $S \leftarrow S$ shift left 1 symbol
            END
    END
    ELSE        % type-4 unit %
    BEGIN
        $b_1 := S_1$
        $S \leftarrow$ shift left 1 bit
        IF ($b_i = 1$) THEN
        BEGIN
            $B \leftarrow$ the first symbol from the left side of $S$
            output $B$ in $f_i$
            $S \leftarrow S$ shift left 1 symbol
        END
    END
END
```