ELSEVIER

Pattern Recognition Letters 20 (1999) 15–20

Pattern Recognition Letters

A note on adaptive 2D-H strings [1]

Ye-In Chang *, Hsing-Yen Ann

Department of Applied Mathematics, National Sun Yat-Sen University, Kaohsiung, Taiwan, ROC

Received 17 April 1998; received in revised form 25 September 1998

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 (Chang and Lin, 1996). We next show how to avoid the ambiguous cases.

Keywords: 2D strings; 2D-H strings; Adaptive 2D-H strings; Image databases; Symbolic pictures

1. Introduction

Chang and Li (1988) have proposed 2D-H strings, which can be viewed as a combination of quadtrees (Samet, 1984, 1990) and 2D strings (Chang et al., 1987). 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, Chang and Lin (1996) 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.

Chang and Lin (1996) 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 the procedure of Chang and Lin 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 (Chang and Lin, 1996). Next, we show how to avoid the ambiguous cases.

* Corresponding author. Tel.: 886 7 5252000 x3819; fax: 886 7 5253809; e-mail: changyi@math.nsysu.edu.tw.

[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.

PII: S 0 1 6 7 - 8 6 5 5 (9 8) 0 0 1 1 6 - 0

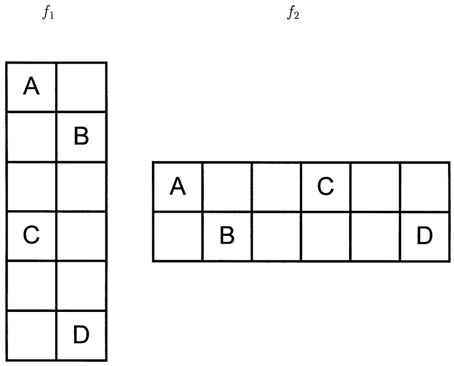

Fig. 1. An example.

2. An ambiguous case

Take Fig. 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 Figs. 2 and 3, respectively.

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

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

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 as adaptive 2D-H strings can be ambiguous. In this example, pictures f_1 and f_2 have the same corresponding quadtree as shown in Fig. 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 Appendix A shows how to reconstruct a picture based on the revised version of the adaptive 2D-H string without causing any ambiguity. 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 if they have the same adaptive 2D-H string representation.

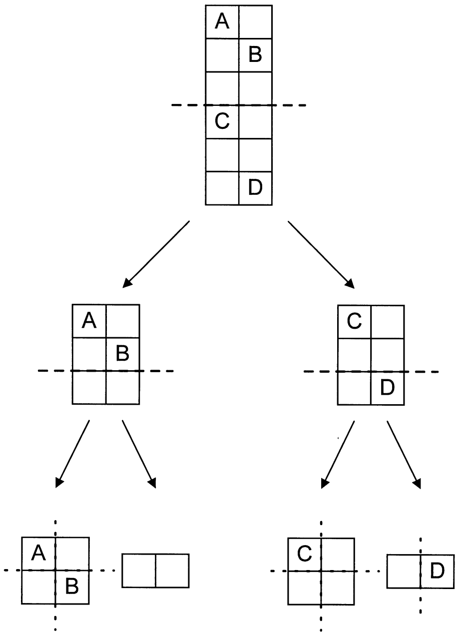

Fig. 2. Decomposition steps for picture f_1.

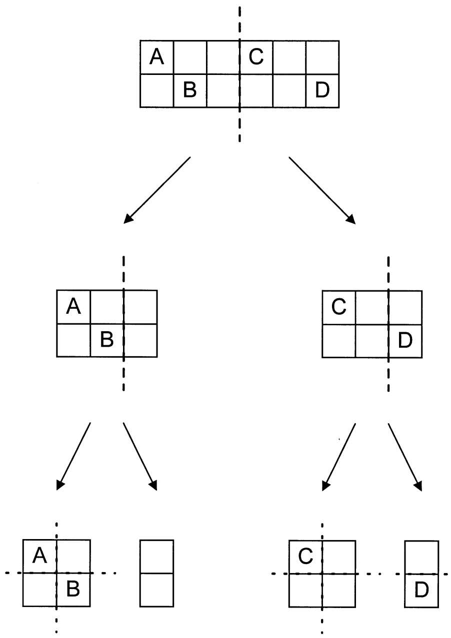

Fig. 3. Decomposition steps for picture f_2.

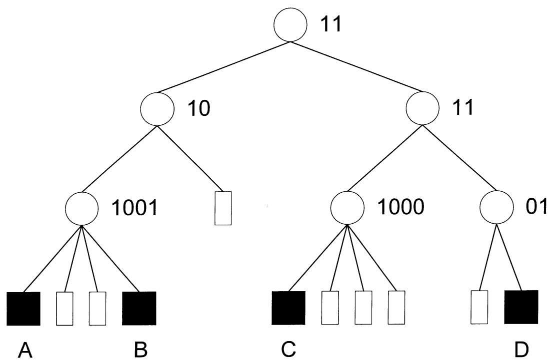

Fig. 4. The quadtree of picture f_1 (f_2).

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.

Appendix A

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 f1, f2, f3 and f4 to be NW, SW, NE and SE
   quadrants subpictures of f, respectively
   % let Si be the ith bit of S from the left side %
   FOR i = 1 to 4
      bi := Si
   S ← S shift left 4 bits
   IF (b1 = 1) THEN
         Reconstruct(f1, ⌈1/2m⌉, ⌈1/2n⌉)      % NW %
     IF (b2 = 1) THEN
         Reconstruct(f2, ⌊1/2m⌋, ⌈1/2n⌉)      % SW %
     IF (b3 = 1) THEN
         Reconstruct(f3, ⌈1/2m⌉, ⌊1/2n⌋)      % NE %
     IF (b4 = 1) THEN
         Reconstruct(f4, ⌊1/2m⌋, ⌊1/2n⌋)      % SE %
END
ELSE IF (m ⩽ 2 and n > 2) THEN      % column segmentation %
```

```
BEGIN
    set f1 and f2 to be W and E quadrant subpictures of f
    FOR i = 1 to 2
      bi := Si
    S ← S shift left 2 bits
    IF (b1 = 1) THEN
      Reconstruct(f1, m, ⌈1/2n⌉)      % W %
    IF (b2 = 1) THEN
      Reconstruct(f2, m, ⌊1/2n⌋)      % E %
END
ELSE IF (m > 2 and n ⩽ 2) THEN      % row segmentation %
BEGIN
    set f1 and f2 to be N and S quadrant subpictures of f
    FOR i = 1 to 2
      bi := Si
    S ← S shift left 2 bits
    IF (b1 = 1) THEN
      Reconstruct(f1, ⌈1/2m⌉, n)      % N %
    IF (b2 = 1) THEN
      Reconstruct(f2, ⌊1/2m⌋, n)      % S %
END
ELSE      % the elementary unit of decomposition %
BEGIN
  IF (m = 2 and n = 2) THEN    % type-1 unit %
  BEGIN
    set f1, f2, f3 and f4 to be NW, SW, NE and SE
    quadrants subpictures of f, respectively
    FOR i = 1 to 4
      bi := Si
    S ← S shift left 4 bits
    FOR i = 1 to 4
      IF (bi = 1) THEN
      BEGIN
        B ← the first symbol from the left side of S
        output B in fi
        S ← S shift left 1 symbol
      END
  END
  ELSE IF (m = 2) THEN      % type-2 unit %
  BEGIN
    set f1 and f2 to be N and S quadrant subpictures of f
    FOR i = 1 to 2
      bi := Si
    S ← S shift left 2 bits
    FOR i = 1 to 2
      IF (bi = 1) THEN
      BEGIN
        B ← the first symbol from the left side of S
```

```
            output B in f_i
            S ← 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 ← S shift left 2 bits
      FOR i = 1 to 2
         IF (b_i = 1) THEN
         BEGIN
            B ← the first symbol from the left side of S
            output B in f_i
            S ← S shift left 1 symbol
         END
   END
   ELSE      % type-4 unit%
   BEGIN
      b_1 := S_1
      S ← shift left 1 bit
      IF (b_i = 1) THEN
      BEGIN
         B ← the first symbol from the left side of S
         output B in f_i
         S ← S shift left 1 symbol
      END
   END
END
```

References

Chang, S.K., Li, Y., 1988. Representation of multi-resolution symbolic and binary pictures using 2D-H strings. In: Proceedings of the IEEE Workshop on Languages for Automata. Maryland, pp. 190–195.

Chang, C.C., Lin, D.C., 1996. A spatial data representation: an adaptive 2D-H string. Pattern Recognition Letters 17 (2), 175–185.

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

Samet, H., 1984. The quadtree and related data structure. ACM Comput. Surveys 16 (2), 187–260.

Samet, H., 1990. Applications of Spatial Data Structure. Addison-Wesley, Reading, MA.