A note on adaptive 2D-H strings

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. © 1999 Elsevier Science B.V. All rights reserved.

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.
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($f_1$)

$= b_N b_S s_N s_S$

$= 11 s_N s_S$

$= 11 10 s_{NN} s_{NS} 11 s_{SN} s_{SS}$

$= 11 10 1001AB 11 1000C 01D$

adaptive 2D-H($f_2$)

$= b_W b_E s_W s_E$

$= 11 s_W s_E$

$= 11 10 s_{WW} s_{WE} 11 s_{EW} s_{EE}$

$= 11 10 1001AB 11 1000C 01D$

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 $f_2$. 
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 \(f\)

1. IF \((\text{min}(m, n) > 2)\) THEN % quadrant segmentation %
2. BEGIN
3. set \(f_1, f_2, f_3\) and \(f_4\) to be NW, SW, NE and SE
4. quadrants subpictures of \(f\), respectively
5. % let \(S_i\) be the \(i\)th bit of \(S\) from the left side %
6. FOR \(i = 1\) to \(4\)
7. \(b_i := S_i\)
8. \(S \leftarrow S\) shift left \(4\) bits
9. IF \((b_1 = 1)\) THEN
10. Reconstruct \((f_1, [1/2m], [1/2n])\) % NW %
11. IF \((b_2 = 1)\) THEN
12. Reconstruct \((f_2, [1/2m], [1/2n])\) % SW %
13. IF \((b_3 = 1)\) THEN
14. Reconstruct \((f_3, [1/2m], [1/2n])\) % NE %
15. IF \((b_4 = 1)\) THEN
16. Reconstruct \((f_4, [1/2m], [1/2n])\) % SE %
17. END
18. ELSE IF \((m \leq 2\) and \(n > 2)\) THEN % column segmentation %

Fig. 4. The quadtree of picture \(f_1\) (\(f_2\)).
19. **BEGIN**
20. set $f_1$ and $f_2$ to be W and E quadrant subpictures of $f$
21. **FOR** $i = 1 \text{ to } 2$
22. $b_i := S_i$
23. $S \leftarrow S$ shift left 2 bits
24. **IF** ($b_1 = 1$) **THEN**
25. Reconstruct($f_1, m, \lceil 1/2n \rceil$) % W %
26. **IF** ($b_2 = 1$) **THEN**
27. Reconstruct($f_2, m, \lceil 1/2n \rceil$) % E %
28. **END**
29. **ELSE IF** ($m > 2$ and $n \leq 2$) **THEN** % row segmentation %
30. **BEGIN**
31. set $f_1$ and $f_2$ to be N and S quadrant subpictures of $f$
32. **FOR** $i = 1 \text{ to } 2$
33. $b_i := S_i$
34. $S \leftarrow S$ shift left 2 bits
35. **IF** ($b_1 = 1$) **THEN**
36. Reconstruct($f_1, m, \lceil 1/2m \rceil, n$) % N %
37. **IF** ($b_2 = 1$) **THEN**
38. Reconstruct($f_2, m, \lceil 1/2m \rceil, n$) % S %
39. **END**
40. **ELSE** % the elementary unit of decomposition %
41. **BEGIN**
42. **IF** ($m = 2$ and $n = 2$) **THEN** % type-1 unit %
43. **BEGIN**
44. set $f_1, f_2, f_3$ and $f_4$ to be NW, SW, NE and SE quadrants subpictures of $f$, respectively
45. **FOR** $i = 1 \text{ to } 4$
46. $b_i := S_i$
47. $S \leftarrow S$ shift left 4 bits
48. **FOR** $i = 1 \text{ to } 4$
49. **IF** ($b_i = 1$) **THEN**
50. **BEGIN**
51. $B \leftarrow$ the first symbol from the left side of $S$
52. output $B$ in $f_i$
53. $S \leftarrow S$ shift left 1 symbol
54. **END**
55. **END**
56. **ELSE IF** ($m = 2$) **THEN** % type-2 unit %
57. **BEGIN**
58. set $f_1$ and $f_2$ to be N and S quadrant subpictures of $f$
59. **FOR** $i = 1 \text{ to } 2$
60. $b_i := S_i$
61. $S \leftarrow S$ shift left 2 bits
62. **FOR** $i = 1 \text{ to } 2$
63. **IF** ($b_i = 1$) **THEN**
64. **BEGIN**
65. $B \leftarrow$ the first symbol from the left side of $S$
66. **END**
References