AD-A058 477 ILLINOIS UNIV AT URBANA-CHAMPAIGN APPLIED COMPUTATION--ETC F/G 9/2
PRESERVING AVERAGE PROXIMITY IN ARRAYS WITH DUPLICATIONS.(U)
APR 78 A L CHOW DAAB07-72-C-0259

UNCLASSIFIED ACT-8 NL

1 OF 1

AD A058 477

END

DATE FILMED 11-78

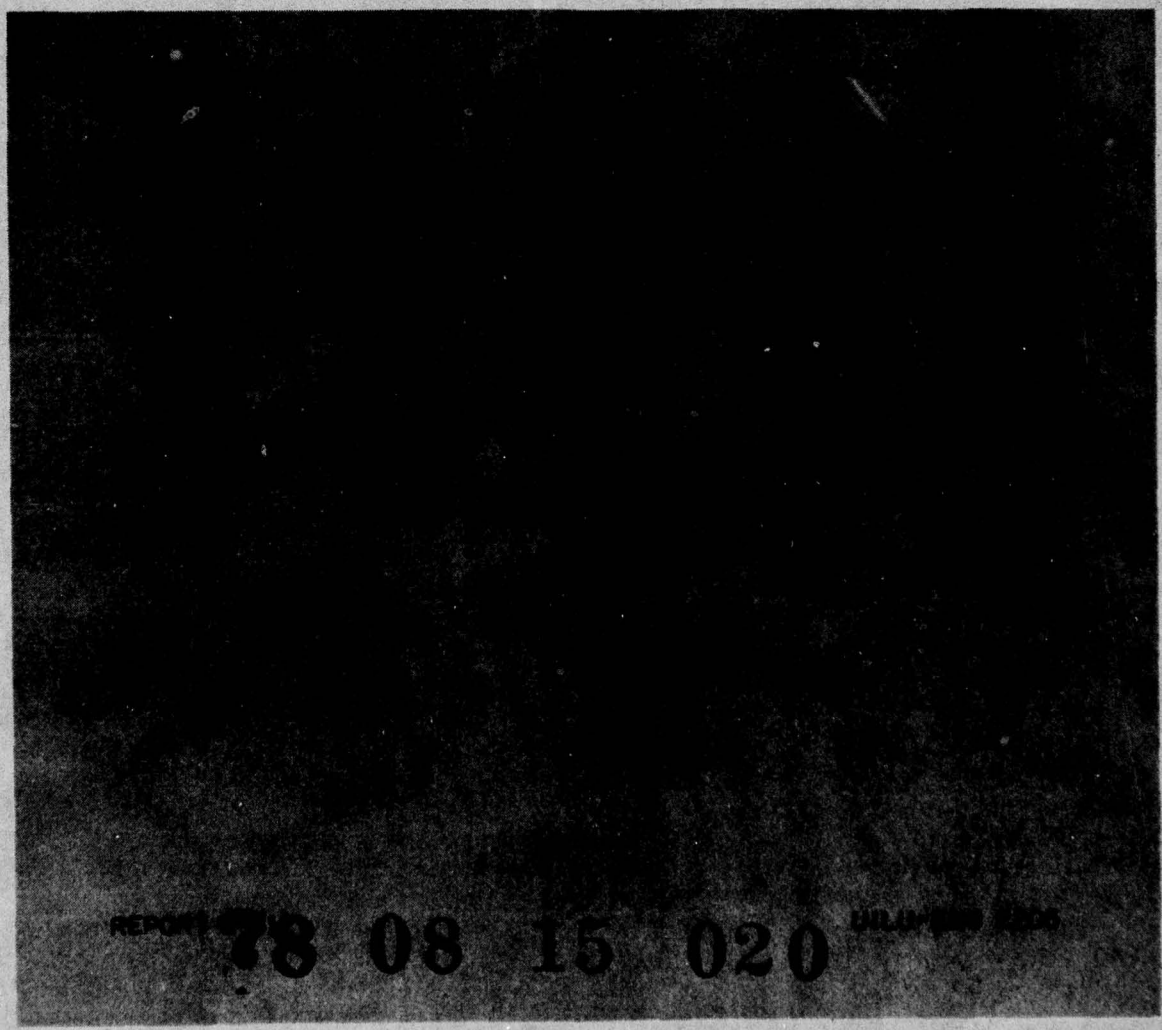
DDC

ACT-8 LEVEL II (12) APRIL, 1978

CSL COORDINATED SCIENCE LABORATORY

APPLIED COMPUTATION THEORY GROUP

PRESERVING AVERAGE PROXIMITY IN ARRAYS WITH DUPLICATIONS



UNIVERSITY OF ILLINOIS – URBANA, ILLINOIS

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) PRESERVING AVERAGE PROXIMITY IN ARRAYS WITH DUPLICATIONS		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER R-812; ACT-8;UILU-ENG 2205
7. AUTHOR(s) Anita Liu Chow		8. CONTRACT OR GRANT NUMBER(s) NSF MCS 76-17321 DAAB-07-72-C-0259
9. PERFORMING ORGANIZATION NAME AND ADDRESS Coordinated Science Laboratory University of Illinois at Urbana-Champaign Urbana, Illinois 61801		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Joint Services Electronics Program		12. REPORT DATE April, 1978 13. NUMBER OF PAGES 57
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Arrays
Array Storage Mappings
Average Proximity
Binary Trees
Redundant Mappings

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Efficiency of storage management in algorithms which use arrays is often enhanced if the arrays are stored in a proximity-preserving manner, that is, array positions which are close to one another in the array are also stored close to one another in the memory structure. It has been shown that any scheme that stores arrays in a linear memory, in both the worst and the average case, induces unbounded loss of proximity, but arrays can be stored in binary trees with bounded loss of average proximity. This paper is devoted to studying the

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. ABSTRACT (continued)

effect of introducing duplication of items of a square array A on the average path length between the images of any two records adjacent to A under mapping from A into the set of leaves of a complete binary tree. It is shown that with the appropriate choice of duplications, in some arrays the average path length can be decreased by as much as 12% without using a deeper tree than needed in the absence of duplication.

(6) PRESERVING AVERAGE PROXIMITY IN ARRAYS WITH DUPLICATIONS.

BY

(10) ANITA LIU CHOW

B.S., University of Illinois, 1976

(14) ACT-8, UILU-ENG-2205

(15) DAAB07-72-C-0259,
NSF-MCS-76-17321

(11) Apr 78

(9) Master's THESIS,

(12) 64p.

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1978.

Thesis Adviser: Professor F. P. Preparata

Urbana, Illinois

PREFACE

Efficiency of storage management in algorithms which use arrays is often enhanced if the arrays are stored in a proximity-preserving manner, that is, array positions which are close to one another in the array are also stored close to one another in the memory structure. It has been shown that any scheme that stores arrays in a linear memory, in both the worst and the average case, induces unbounded loss of proximity, but arrays can be stored in binary trees with bounded loss of average proximity. This paper is devoted to studying the effect of introducing duplication of items of a square array A on the average path length between the images of any two records adjacent in A under a mapping from A into the set of leaves of a complete binary tree. It is shown that with the appropriate choice of duplications, in some arrays the average path length can be decreased by as much as 12% without using a deeper tree than needed in the absence of duplication.

ACKNOWLEDGEMENT

I would like to express my gratitude and appreciation to my advisor, Professor Franco Preparata, for his advice and support. In addition, I thank Mrs. Phyllis Young for typing this thesis.

Finally, I would like to thank my parents and my husband for their encouragement and support.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The running time of an algorithm is usually measured by summing the execution time attached to each operation performed. In practice, the running time, or complexity, of an algorithm depends very much on how efficiently its data structure can be accessed. Algorithms which operate on arrays frequently access the array in accordance with a notion of locality, that is, the array cell accessed after cell v is accessed is in a neighborhood of v. For instance, the usual matrix multiplication traverses rows and columns of the matrices; and the Strassen's matrix multiplication [1] accesses matrices in blocks of four. Therefore, it is important to store the array in a manner so that the proximity is preserved, that is, cells that are close to one another in the array are also stored close to one another in the memory structure.

Preservation of proximity in arrays has been studied in [2], [3], and [4]. In [2], A. Rosenberg showed that any scheme that stores arrays in a linear memory, in the worst case, induces unbounded loss of proximity. In [4], R. A. DeMillo et al showed that even average loss of proximity is unbounded in such schemes. However, they showed that arrays can be stored in binary trees with bounded loss of average proximity.

In this paper, I will discuss the effect of introducing duplication of items of an $n \times n$ array A on the average path length between the images of any two records adjacent in A under a mapping from A into the set of leaves of a complete binary tree. Arrays will be represented as graphs: vertices represent array cells containing records, and arcs represent logical adjacencies.

This paper consists of six chapters. In the next chapter, graphical models of arrays with and without duplications are defined. In Chapter 3, a binary tree memory structure is defined and a storage allocation mapping is presented. In Chapter 4, a method for finding the stationary probability distribution of any array is presented and the average stretch between two vertices is formulated. In Chapter 5, various duplicating patterns are discussed and an efficient duplicating scheme is outlined. Finally, conclusions are drawn.

CHAPTER 2

GRAPHICAL MODELS OF AN n × n ARRAY

2.1. With No Duplication

An n × n array is a two-dimensional data structure. It consists of n rows and n columns of cells. Each cell can hold one record. If each cell of an array holds a distinct record, then the array is said to be with no duplication. In this section, an array with no duplicates will be modeled as a directed graph.

A finite directed graph G = (V,E) consists of a finite set V of vertices and a set E of ordered pairs of vertices, the arcs. (v,w) is an arc directed from a vertex v to a vertex w and the direction is indicated by an arrow-head on the arc. A vertex w is called a successor of a vertex v if there is an arc (v,w); vertex v is then called the predecessor of w. The set of successors of a vertex v will be denoted by Succ(v) and the set of predecessors of v will be denoted by Pred(v).

Let I(n) denote the index set from 1 to n. Suppose A is an n × n array with no duplication; it consists of n^2 cells $c_{i,j}$, i,j ∈ I(n) where i denotes the row and j denotes the column. Then the graphical model G_A of A is a directed graph (V,E), where $V = \{c_{i,j} \mid i,j \in I(n)\}$ and $E = \{(c_{i,j},c_{i,j+1})(c_{i,j+1},c_{i,j}),(c_{i,j},c_{i+1,j}),(c_{i+1,j},c_{i,j}) \mid i,j \in I(n-1)\}$. The graph G_A of the 3 × 3 array A with no duplication is shown in Figure 1.

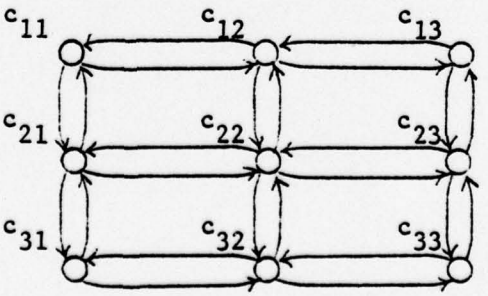


Figure 1. The graph G_A of the 3 × 3 array A with no duplication.

2.2. With Duplications

Duplicate copies of a record can be introduced into an array to enhance its performance. When n is not a power of 2, an $n \times n$ array can be padded to an $2^{\lceil \log n \rceil} \times 2^{\lceil \log n \rceil}$ [1] array with duplicate copies. A systematic scheme will be described.

A $2^s \times 2^s$ array can be divided into smaller square arrays by a set of horizontal and vertical lines. These lines are called j^{th} boundaries, for some $j \in I(s-1)$, they are defined recursively as follows:

1. the $(s-1)^{th}$ boundaries divide an $2^s \times 2^s$ array into four $2^{s-1} \times 2^{s-1}$ subarrays;
2. the j^{th} boundaries divide each of the $2^{2(s-j-1)}$ $2^{j+1} \times 2^{j+1}$ arrays into four $2^j \times 2^j$ subarrays.

In figure 2, $(s-1)^{th}$ boundaries and $(s-2)^{th}$ boundaries of the $2^s \times 2^s$ array are denoted by boldface lines and dotted lines, respectively. The i^{th} boundary is said to be higher than the j^{th} boundary if i is larger than j.

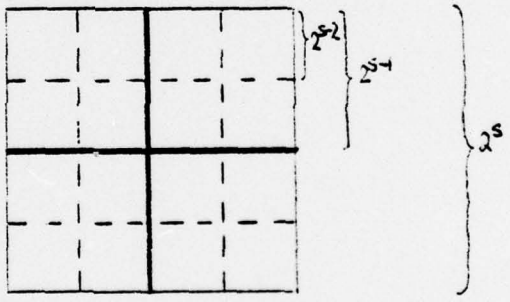


Figure 2. $(s-1)^{th}$ (boldface) and $(s-2)^{th}$ (dotted) boundaries of an $2^s \times 2^s$ array.

---

[1] All logarithms are to the base two.

A cell is said to be at the vertical (horizontal) j^{th} boundary, $j > 0$, if it is in column (row) $2^j(2i-1) + k$, for some $i \in I(2^{s-j-1})$ and some $k \in \{0,1\}$. Only the cells in columns (rows) 1 and 2^s are said to be at the vertical (horizontal) 0^{th} boundaries.

A finite sequence of nonnegative integers $(a_1,a_2,\ldots)$ is called a duplicating pattern if a_j is the number of duplications at each j^{th} boundary. The duplicating patterns suggest a systematic duplication at the boundaries. Suppose A is an $n \times n$ array, $n = 2^s+t$, $s \geq 1$ and $1 \leq t \leq 2^s$, and suppose each cell $c_{i,j}$ of A contains a record $r_{i,j}$, $i,j \in I(n)$. Then the extended array A* of A under the duplicating pattern $(a_1,a_2,\ldots,a_s)$ is defined recursively as follows. If $s = 0$, A* is A. Otherwise, A* is an $2^{s+1} \times 2^{s+1}$ array consisting of four $2^s \times 2^s$ subarrays A^*_{11}, A^*_{12}, A^*_{21} and A^*_{22}. Each $A^*_{i,j}$ is the extended array of $A_{i,j}$ under the duplicating pattern $(a_1,a_2,\ldots,a_{s-1})$ and $A_{i,j}$ is an $n' \times n'$ array, where $n' = (n+a_s)/2$, containing records $r_{k\ell}$, $k = (i-1)n''+1,(i-1)n''+2,\ldots,(i-1)n''+n'$, $\ell = (j-1)n''+1,\ldots,(j-1)n''+n'$, where $n''=(n-a_s)/2$. The construction of the extend array of a 5×5 array under the duplicating pattern (1,1) is illustrated in figure 3.

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} \\ r_{21} & r_{22} & r_{23} & r_{24} & r_{25} \\ r_{31} & r_{32} & r_{33} & r_{34} & r_{35} \\ r_{41} & r_{42} & r_{43} & r_{44} & r_{45} \\ r_{51} & r_{52} & r_{53} & r_{54} & r_{55} \end{bmatrix}$$

(a) an 5×5 array A

$$A_{11} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \qquad A_{12} = \begin{bmatrix} r_{13} & r_{14} & r_{15} \\ r_{23} & r_{24} & r_{25} \\ r_{33} & r_{34} & r_{35} \end{bmatrix}$$

$$A_{21} = \begin{bmatrix} r_{31} & r_{32} & r_{33} \\ r_{41} & r_{42} & r_{43} \\ r_{51} & r_{52} & r_{53} \end{bmatrix} \qquad A_{22} = \begin{bmatrix} r_{33} & r_{34} & r_{35} \\ r_{43} & r_{44} & r_{45} \\ r_{53} & r_{54} & r_{55} \end{bmatrix}$$

(b) 3×3 arrays A_{ij}

$$(A_{11})_{11} = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} \qquad (A_{11})_{12} = \begin{bmatrix} r_{12} & r_{13} \\ r_{22} & r_{23} \end{bmatrix}$$

$$(A_{11})_{21} = \begin{bmatrix} r_{21} & r_{22} \\ r_{31} & r_{32} \end{bmatrix} \qquad (A_{11})_{22} = \begin{bmatrix} r_{22} & r_{23} \\ r_{32} & r_{33} \end{bmatrix}$$

(c) $(A_{11})_{ij} = (A_{11})^*_{ij}$ ($(A_{12})_{ij}, (A_{21})_{ij}, (A_{22})_{ij}$, $i,j \in I(2)$ are constructed similarly)

$$A^*_{ij} = \begin{bmatrix} (A_{ij})^*_{11} & (A_{ij})^*_{12} \\ (A_{ij})^*_{21} & (A_{ij})^*_{22} \end{bmatrix}$$

(d) extended array A^*_{ij} of A_{ij} under duplicating pattern (1)

$$A^* = \begin{bmatrix} A^*_{11} & A^*_{12} \\ A^*_{21} & A^*_{22} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{12} & r_{13} & r_{13} & r_{14} & r_{14} & r_{15} \\ r_{21} & r_{22} & r_{22} & r_{23} & r_{23} & r_{24} & r_{24} & r_{25} \\ r_{21} & r_{22} & r_{22} & r_{23} & r_{23} & r_{24} & r_{24} & r_{25} \\ r_{31} & r_{32} & r_{32} & r_{33} & r_{33} & r_{34} & r_{34} & r_{35} \\ r_{31} & r_{32} & r_{32} & r_{33} & r_{33} & r_{34} & r_{34} & r_{35} \\ r_{41} & r_{42} & r_{42} & r_{43} & r_{43} & r_{44} & r_{44} & r_{45} \\ r_{41} & r_{42} & r_{42} & r_{43} & r_{43} & r_{44} & r_{44} & r_{45} \\ r_{51} & r_{52} & r_{52} & r_{53} & r_{53} & r_{54} & r_{54} & r_{55} \end{bmatrix}$$

(e) extended array A* of A under (1,1)

Figure 3. Construction of the extended array A* of a 5 × 5 array A under the duplicating pattern (1,1).

It is assumed that the probability of accessing a cell c_{ij} containing record r_{ij} in the array A is equal to the sum of the probabilities of accessing the cells containing the record r_{ij} in the extended array A*.

The graphical model G_{A^*} of an extended array A* of an $2^s+t \times 2^s+t$ array A under the duplicating pattern $(a_1,a_2,\ldots,a_s)$ is a directed graph (V,E), $V = \{c^*_{i,j} \mid i,j \in I(2^{s+1})\}$ and $E = \{(c^*_{i,j},c^*_{i,k_j}),(c^*_{i,j+1},c^*_{i,\ell_{j+1}}) \mid i,j \in I(2^{s+1}-1)$ and k_j is the smallest integer larger than j such that c^*_{i,k_j} contains $r_{x,y+1}$ if $c^*_{i,j}$ contains $r_{x,y}$. ℓ_j is the largest integer smaller than j such that c^*_{i,ℓ_j} contains $r_{x,y-1}$ if $c_{i,j}$ contains $r_{x,y}\}$

$\cup \{(c^*_{i,j}, c^*_{k_i,j}), (c^*_{i+1,j}, c^*_{\ell_{i+1},j}) \mid i,j \in I(2^{s+1}-1)$ and k_i is the smallest integer larger than i such that $c^*_{k_i,i}$ contains $r_{x+1,y}$ if $c^*_{i,j}$ contains $r_{x,y}$ and ℓ_i is the largest integer smaller than i such that $c^*_{\ell_i,j}$ contains $r_{x-1,y}$ if $c^*_{i,j}$ contains $r_{x,y}\}$.

In the graphical models as defined above, there are two classes of connections, the horizontal connections which connect vertices in the same row and the vertical connections which connect vartices in the same column. Furthermore, all rows have the same horizontal connections and all the columns have the same vertical connections. The graphical model G^*_A of the extended array A* constructed in figure 3 is shown in figure 4.

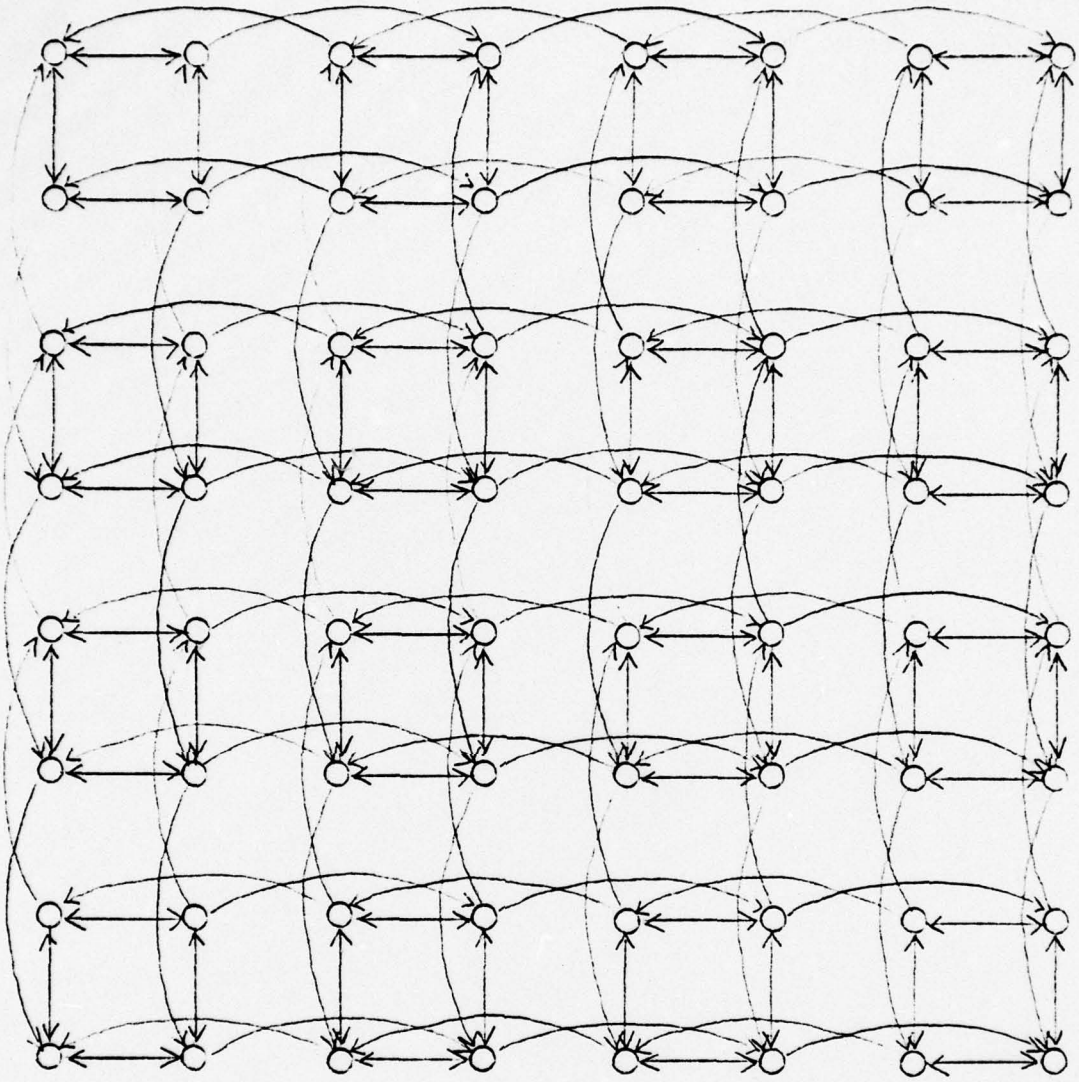


Figure 4. Graphical model G_{A^*} of A* in Figure 3(e).

CHAPTER 3

BINARY TREE MEMORY ORGANIZATION

3.1. Memory Configuration

There is a distinction between linear memory and arbitrary lists [4] in array storage mapping, namely arrays can be stored as nonlinear lists with bounded loss of average proximity, but cannot be stored in a linear memory. In this paper, binary tree memories will be used as the storage structure.

A binary tree memory structure, BTMS, is hierarchically organized. It is either a single memory cell or it consists of a complete binary tree of four sub-BTMSs of equal size. Each memory cell can hold one record. The level of a BTMS which consists of a memory cell is zero, otherwise the level of a BTMS is 2+ level of one of its sub-BTMSs. BTMSs with zero, two, four levels are shown in figure 5.

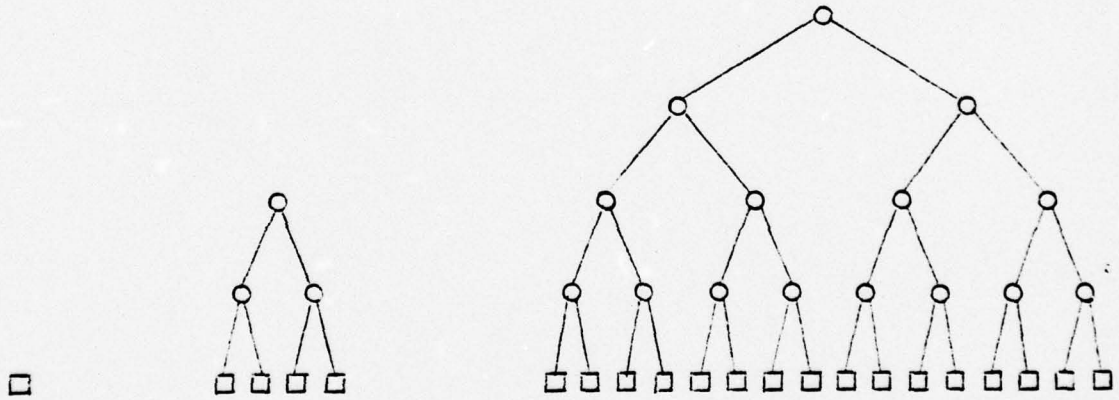


Figure 5. BTMSs with 0, 2 and 4 levels.

In the figure, the circles denote linkage cells and the squares denote data cells.

3.2. Array Storage Mapping

The binary tree memory BTMS is described in the last section. A recursive method of mapping an array of records into memory cells of a BTMS will be described in this section.

Consider an $n \times n$ extended or non-extended array A, where $n = 2^s$ and divide A along the $(s-1)^{th}$ boundaries to form four $n/2 \times n/2$ subarrays A_{11}, A_{12}, A_{21}, and A_{22}. Suppose A is to be stored in a BTMS B which consists of four sub-BTMSs B_1, B_2, B_3, B_4 from left to right at the second level of the hierarchy. Then A_{11}, A_{12}, A_{21}, and A_{22} are stored in B_1, B_2, B_3, and B_4, respectively. The process is continued recursively until the subarrays are of order 1×1 which results in mapping a record into a memory cell.

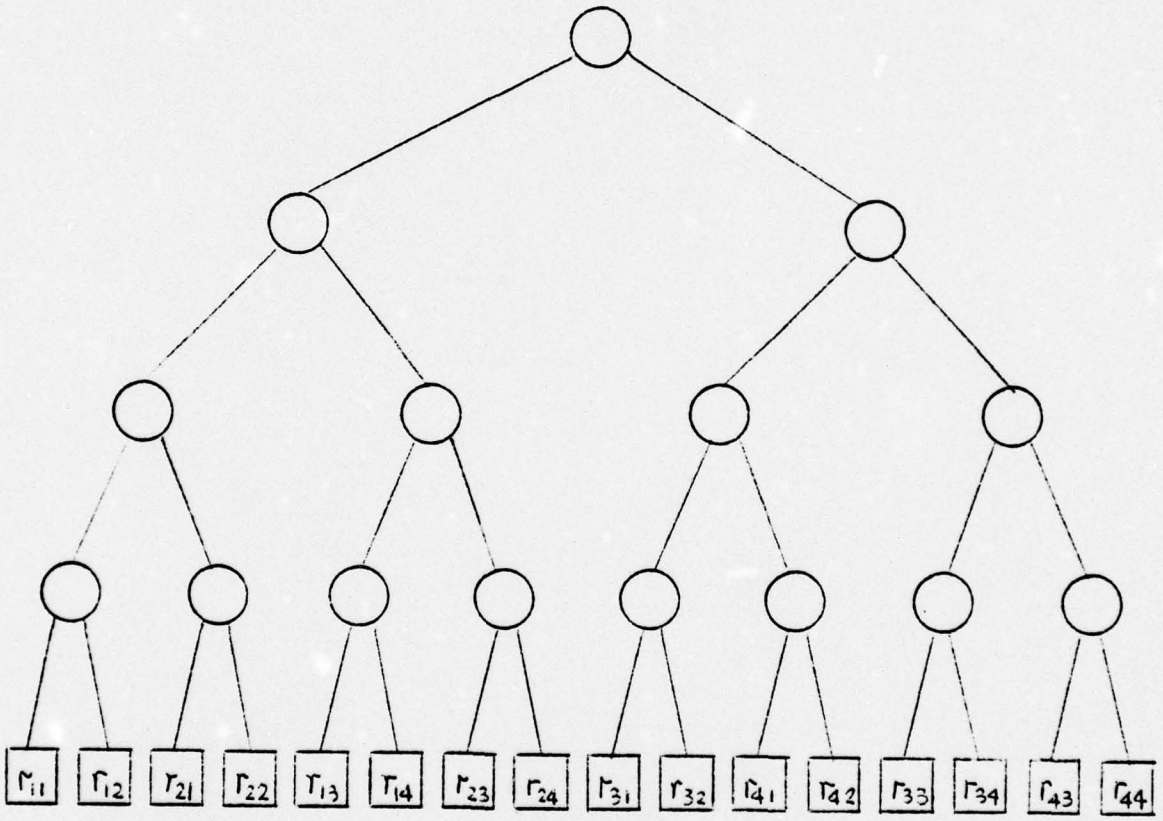
Figure 6 shows the results of mapping an 4×4 array A of records r_{ij}, $i,j \in I(4)$, into the memory cells of a BTMS with four levels.

With the graphical models described in Chapter 2, storing an $2^s \times 2^s$ array A in a 2s-level BTMS is represented as mapping the set of vertices of the graph G_A into the leaves of a complete binary tree of height 2s.

When n is not a power of two, the storage mapping is applied to an $2^{\lceil \log n \rceil} \times 2^{\lceil \log n \rceil}$ array and the records of the original array are held in the first n rows and first n columns in the top left corner of the new array. The remaining cells are empty.

$$A = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} \\ r_{21} & r_{22} & r_{23} & r_{24} \\ r_{31} & r_{32} & r_{33} & r_{34} \\ r_{41} & r_{42} & r_{43} & r_{44} \end{bmatrix}$$

(a) an 4 x 4 array A



(b) 4-levels BTMS containing A

Figure 6. Array storage mapping.

CHAPTER 4

STRETCHES UNDER THE STORAGE MAPPING

4.1. Definition of Stretch

Under the storage mapping defined in the previous chapter, the image of an arc (v,w) in G_A is defined as the path from the image of vertex v to the image of vertex w in the BTMS. The ratio of the length of the image of an arc (v,w) to the length of the arc (v,w) is called the stretch of (v,w) under the mapping. All arcs have length one, hence the stretch of an arc is an integer equal to the path length of the image of the arc. It will be shown in the following lemma that the stretch of an arc is completely determined by the position of the arc in the graph.

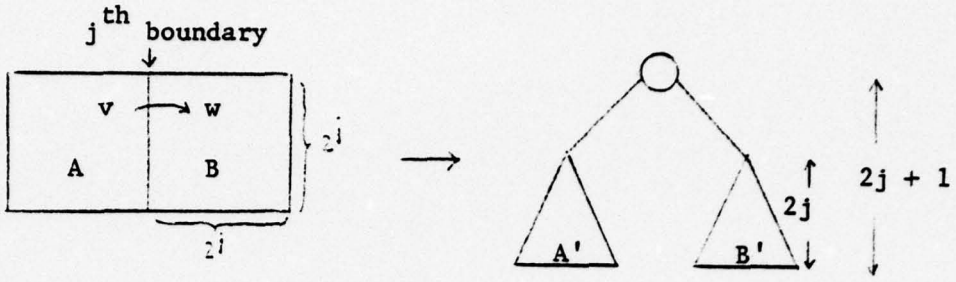
Lemma 4.1.

Let the vertical j^{th} (horizontal i^{th}) boundary be the highest boundary that an arc e crosses in the graph. The stretch of the arc e is $4j + 2$ $(4i + 4)$.

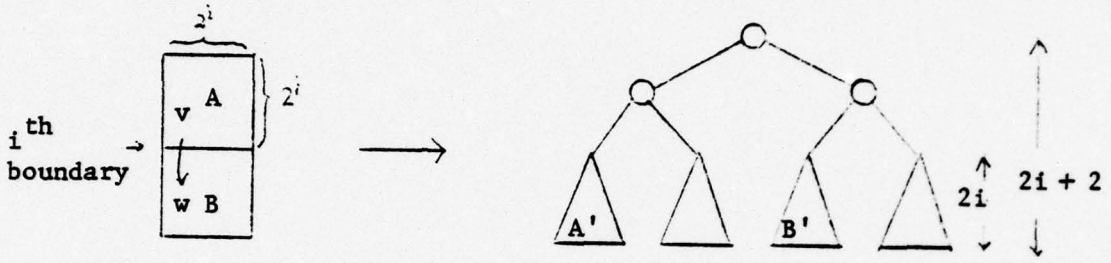
Proof

Consider a horizontal arc (v,w) in G_A (refer to Figure 7(a)) and j^{th} boundary is the highest vertical boundary that (v,w) crosses. Then v and w are in two horizontally adjacent $2^j \times 2^j$ arrays and their images under the mapping will be in the left subtree and the right subtree of a BTMS of level $2j + 1$. Therefore, the stretch of (v,w) is $4j + 2$.

For a vertical arc (v,w), v and w are in two vertically adjacent $2^i \times 2^i$ arrays if the horizontal i^{th} boundary is the highest boundary that the arc (v,w) crosses. The images of v and w under the mapping are in the left subtree and the right subtree of a BTMS of level $2i + 2$. Therefore, the stretch of (v,w) is $4i + 4$. □



(a) Stretch of a horizontal arc



(b) Stretch of a vertical arc

Figure 7. Illustration of the proof of Lemma 4.1.

4.2. Average Stretch

4.2.1. Probability distribution of the vertices of the array

The collection of sequences of visits to the vertices of the graph G_A is a stochastic process. We assume that this process is a random walk [5], that is the conditional probability of traversing an arc originated at a vertex v is a constant which is equal to the reciprocal of the cardinality of Succ(v), and the moves are independent. Furthermore, the graph G_A is irreducible.

In an array of records r_{ij}, $i,j \in \{0,1,\ldots,n-1\}$, record r_{ij} is even if $(i+j) \bmod 2 = 0$ and is odd otherwise. A vertex is called even (odd) if it represents a cell containing an even (odd) record. In a graph model $G_A = (V,E)$, the vertices set V can be partitioned into two sets: V_1 consists of odd vertices and V_2 consists of even vertices. At even time units, only vertices in a particular set V_1 or V_2 can be accessed, without loss of generality say the set is V_2; so at odd time units, only vertices in V_1 can be accessed. Hence, from the graph $G_A = (V,E)$, we can obtain two disjoint graphs $G_1 = (V_1,E_1)$ and $G_2 = (V_2,E_2)$ where $E_i = \{(v,w) \mid v,w \in V_i$ and there is a path from v to w of length 2 in $G_A\}$ for $i = 1,2$. Let $\text{Pred}_i(v)$ be the set of predecessors of the vertex v in G_i.(*) Each of G_1 and G_2 is an irreducible and aperiodic Markov process. Therefore, we have the following lemma.

Lemma 4.2.

For a given graph $G_A = (V,E)$ of an array A, there exist two unique limiting (steady-state) probability distributions p_1 and p_2 of V such that at odd time units, p_1 satisfies the following set of equations:

$$p_1(v) = \sum_{w \in \text{Pred}_1(v)} p_1(w)\text{Pr}_1(v|w) \text{ for all } v \in V_1,$$

$$p_1(v) = 0 \text{ for all } v \in V_2,$$

and

$$\sum_{v \in V} p_1(v) = 1;$$

at even time units, p_2 satisfies the following set of equations:

(*) Also for $v, w \in V_i$, with paths of length 2 between v and w in G_A, let $\text{Pr}_i(v|w)$ denote the probability of reaching v from w in two time units.

$$p_2(v) = \sum_{w \in \text{Pred}_2(v)} p_2(w)\Pr_2(v|w) \text{ for all } v \in V_2$$

$$p_2(v) = 0 \text{ for all } v \in V_1,$$

and

$$\sum_{v \in V} p_2(v) = 1.$$

Since a randomly selected time unit is odd or even with equal probabilities, we can take the arithmetic mean $\bar{p}$ of p_1 and p_2 as the stationary distribution at all times. A stationary probability distribution p of V of $G_A = (V,E)$ is one which satisfies the following equations:

$$p(v) = \sum_{w \in \text{Pred}(v)} p(w)\Pr(v|w), \text{ for all } v \in V \qquad (4.1)$$

and

$$\sum_{v \in V} p(v) = 1.$$

Lemma 4.3.

The distribution $\bar{p} = (p_1 + p_2)/2$ of V is stationary.

Proof:

Since p_1 and p_2 are steady-state distribution at odd and even time units, respectively, then

$$\sum_{\substack{w \in V \\ w \in \text{Pred}(v)}} p_2(w)\Pr(v|w) = \sum_{\substack{w \in V_1 \\ w \in \text{Pred}(v)}} p_2(w)\Pr(v|w) + \sum_{\substack{w \in V_2 \\ w \in \text{Pred}(v)}} p_2(w)\Pr(v|w)$$

But the first term is 0, since $p_2(w) = 0$ when $w \in V_1$, and the second term is equal to $p_1(v)$, thus

$$\text{(i)} \qquad p_1(v) = \sum_{\substack{w \in V \\ w \in \text{Pred}(v)}} p_2(w)\Pr(v|w).$$

Similarly we obtain

$$\text{(ii)} \qquad p_2(v) = \sum_{\substack{w \in V \\ w \in \text{Pred}(v)}} p_1(w)\Pr(v|w).$$

Summing (i) and (ii) and dividing both sides by 2, we obtain

$$\frac{p_1(v) + p_2(v)}{2} = \sum_{\substack{w \in V \\ w \in \text{Pred}(v)}} \frac{p_1(w) + p_2(w)}{2} \Pr(v|w)$$

which satisfies equation (4.1). □

Since it is assumed that the conditional probability of traversing an arc is a constant, the stationary probability distribution of the set of arcs can be easily obtained from the stationary probability distribution of the set of vertices. The stationary probability distribution p(i), $i \in I(n^2)$, can be expressed as the product of a positive weight w(i) and a normalization factor equal to the reciprocal of the total weight of all the vertices. Therefore, equation 4.1 can be written as follows.

$$w(i) = \sum_{j \in \text{Pred}(i)} w(j)\Pr(i|j) \text{ for all } i \in V \qquad (4.2)$$

The graph G_A of a square array A with no duplication is highly symmetrical. Hereafter we let (i,j) denote the vertex at the intersection of row i and column j in A. The following lemma gives the stationary weight distribution for the vertices.

Lemma 4.4.

The vertices of the graph $G_A = (V,E)$, $V = \{(i,j) \mid i,j \in I(n)\}$ have the following weight distribution: (Figure 8)

$w(1,1) = w(1,n) = w(n,1) = w(n,n) = 2$

$w(1,j) = w(i,1) = w(n,j) = w(i,n) = 3$ for all $i,j = 2,3, \ldots, n-1$

$w(i,j) = 4$, for all $i,j = 2,3, \ldots, n-1$

and the normalization factor is $4n(n-1)$.

Proof:

It is required to show that for every vertex $(i,j) \in V$, equation (4.2) is satisfied by the weight distribution given in the lemma.

For $(i,j) = (1,1)$, $(n,1)$, $(1,n)$ and (n,n), $w(i,j) = 3 \cdot \frac{1}{3} + 3 \cdot \frac{1}{3} = 2$; so the equation is satisfied. For other vertices, it can be showed analogously that the equations (4.2) are satisfied.

The normalization factor is the sum of all the weights, $\sum_{i,j \in I(n)} w(i,j)$, which is equal to $4n(n-1)$. □

2	3	3	...	3	2
3	4	4	...	4	3
3	4	4	...	4	3
.	.	.	...	.	.
.	.	.	...	.	.
.	.	.	...	.	.
3	4	4	...	4	4
2	3	3	...	3	2

Figure 8. Weight distribution of an array with no duplication.

From Lemma 4.4 and the definition of weight distribution, we have the following theorem.

Theorem 4.1.

Let A be an $n \times n$ array with no duplication. Then the vertices in set $V = \{(i,j) \mid i,j \in I(n)\}$ of the graph G_A has the following stationary probability distribution:

$p(1,j) = p(i,1) = 2/\phi$ for $i,j = 1,n$,

$p(1,j) = p(n,j) = p(i,1) = p(i,n) = 3/\phi$ for $i,j = 2,3,\ldots,n-1$,

$p(i,j) = 4/\phi$ for $i,j = 2,3,\ldots,n-1$,

where $\phi = 4n(n-1)$.

Corollary 4.1

The arcs of E in G_A, with no duplication are equiprobable.

Proof:

For each arc $(v,w) \in E$,

$Pr((v,w)) = p(v) \cdot Pr(w|v)$

and $Pr(w|v) = \begin{cases} \frac{1}{2} & \text{if v is at positions } (1,1),(1,n),(n,1), \text{ or } (n,n) \\ \frac{1}{3} & \text{if v is at position } (1,j),(n,j),\ (i,1),(i,n) \text{ for } 2 \le i,j \le n-1 \\ \frac{1}{4} & \text{otherwise.} \end{cases}$

Thus, by Theorem 4.1, $Pr((v,w)) = \frac{1}{\phi}$ for all edges $(v,w) \in E$. □

The graph G_A^* of the extended array A* of a square array A is more complicated. The following lemmas are established for finding the stationary probability distribution of the set of vertices of G_A^*.

Lemma 4.5.

For a graph $G_A^* = (V,E)$, $V = \{(i,j) \mid i,j \in I(2^{s+1})\}$ of a $2^{s+1} \times 2^{s+1}$ extended array A*, we have the following relations among the terms of w, the stationary weight distribution of V, (Figure 9)

$$w(i,j) = \begin{cases} w(j,i) \\ w(j,2^{s+1} + 1 - i) \\ w(i,2^{s+1} + 1 - j) \\ w(2^{s+1} + 1 - i,\ 2^{s+1} + 1 - j) \\ w(2^{s+1} + 1 - j,\ 2^{s+1} + 1 - i) \\ w(2^{s+1} + 1 - j, i) \\ w(2^{s+1} + 1 - i, j) \end{cases} \quad \text{for } i,j \in I(2^s)$$

Proof:

Observe that the graph G_A^* is symmetrical (by definition) about the horizontal s^{th} boundary, the vertical s^{th} boundary, and the main diagonal. □

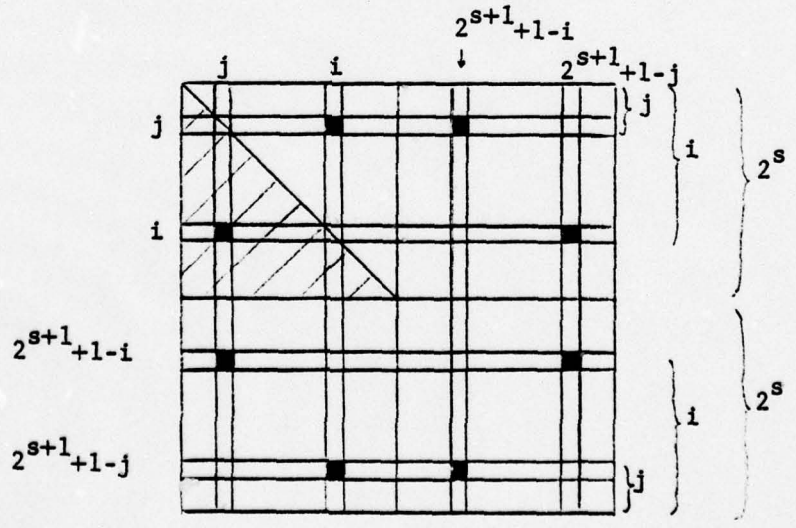


Figure 9. Symmetries of the weight distribution (the vertices at the marked position have identical weight).

From Lemma 4.5, we only have to find the weights of the vertices (i,j) for $i = 1,2,\ldots,2^s$ and $j = 1,2,\ldots,i$, (the shaded triangle in Figure 9) rather than those corresponding to all $(2^{s+1})^2$ positions.

Lemma 4.6.

Let $w(i,2^s)$ and $w(2^2,j)$, $1 \le i, j \le 2^s$, be given. The set of weights

$$w(1,1) = \frac{8}{9}\, w(1,2^s) \cdot w(2^s,1)$$

$$w(i,j) = w(i,2^s) \cdot w(2^s,j) \text{ for } 1 \le i,j \le 2^s \text{ and } (i,j) \neq (1,1),$$

and $w(i,j)$ as in Lemma 4.5 for i or $j > 2^s$ satisfies equation (4.2) (refer to Figure 10).

Proof:

Observe that $w(2^s,j) = w(j,2^s)$ due to the symmetry of the graph G_A^* about the diagonal. Assuming $(i,j) \neq (1,1)$, it is required to show that $w(i,j) = w(2^s,j)$ satisfies equation (4.2), which is

$$w(i,j) = \sum_{(i,k)\in Pred(i,j)} w(i,k)\ Pr((i,j)\,|\,(i,k)) + \sum_{(k,j)\in Pred(i,j)} w(k,j)\ Pr((i,j)\,|\,(k,j)).$$

Recall that all the rows have the same horizontal connections and all the columns have the same vertical connections in G_A^* and let C_i be $\frac{4}{3}$ if $i \in \{1,2^{s+1}\}$ and 1 otherwise. Hence, the right hand side of the equation can be expressed as

$$w(2^s,i) \sum_{(2^s,k)\in Pred(2^s,j)} w(2^s,k) \cdot C_i \cdot Pr((2^s,j)\,|\,(2^s,k))$$

$$+ w(2^s,j) \sum_{(k,2^s)\in Pred(i,2^s)} w(k,2^s) C_j Pr((i,2^s)\,|\,(k,2^s)). \quad (*)$$

$$\text{Since } w(2^s,2^s) = \sum_{(2^s,k)\in Pred(2^s,2^s)} w(2^s,k)Pr((2^s,2^s)|(2^s,k)) + \sum_{(k,2^s)\in Pred(2^s,2^s)} w(k,2^s)Pr((2^s,2^s)|(k,2^s))$$

and (due to the symmetry around the main diagonal) the two terms in the right side are equal, $w(2^s,2^s) = 2 \sum_{(k,2^s)\in Pred(2^s,2^s)} w(k,2^s)Pr((2^s,2^s)|(k,2^s))$.

Furthermore, the definition of $w(i,j)$ implies that $w(2^s,2^s) = 1$. Letting $i = 2^s$ in the expression (*) we have,

$$w(2^s,j) = w(2^s,2^s) \sum_{(2^s,k)\in Pred(2^s,j)} w(2^s,k)C_{2^s}Pr((2^s,j)|(2^s,k)) + w(2^s,j) \sum_{(k,2^s)\in Pred(2^s,2^s)} w(k,2^s)C_j Pr((2^s,2^s)|(k,2^s))$$

Hence,

$$w(2^s,j) = 1/(1-C_j/2) \sum_{(2^s,k)\in Pred(2^s,j)} w(2^s,k)Pr((2^s,j)|(2^s,k))$$

Substituting the identity just obtained into expression (*), we have

$$w(i,j) = \begin{cases} w(2^s,i) \cdot w(2^s,j) & \text{for } 1 \le i,j \le 2^s \text{ and } (i,j) \ne (1,1) \\ \frac{8}{9} w(2^s,1) \cdot w(2^s,1) & \text{for } i,j = 1. \end{cases}$$

□

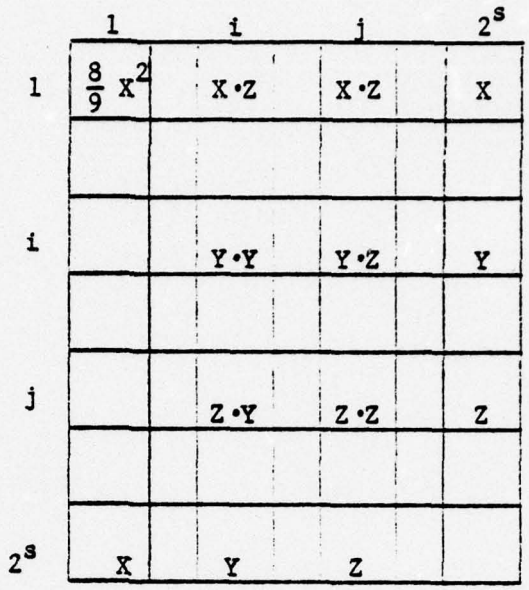


Figure 10. Weight distribution in upper left quadrant of a $2^{s+1} \times 2^{s+1}$ array.

From Lemmas 4.5 and 4.6, we only have to find the weights $w(2^s,j)$, denoted as $\bar{w}(j)$, for $j = 1,\ldots,2^s$, rather than those corresponding to all $(2^{s+1})^2$ positions. For an extended array of an $(2^s+t) \times (2^s+t)$ array under the duplicating pattern $(a_1,a_2,\ldots,a_s)$, the weights $\bar{w}(j)$ will be found by a sequence of s iterations. In the initial step, all a_i except a_s are assumed to be zero and we find the weights $\bar{w}^s(j)$ for the pattern $(0,0,\ldots,0,a_s)$. In the next step, we extend the pattern to include a_{s-1}, and find $\bar{w}^{s-1}(j)$, for $(0,0,\ldots,0,a_{s-1},a_s)$, in terms of $\bar{w}^s(j)$. This process is continued until the pattern is extended to including all a_i's: at this point, the weights $\bar{w}^1(j)$ are the $\bar{w}(j)$, for $j = 2,\ldots,2^s$ and $\bar{w}(1) = \frac{3}{4}\bar{w}^1(1)$. The weights $\bar{w}^i(j)$ are generated in each iteration i as follows. Initially, we have

$$\bar{w}^s(j) = a_s + 1 \quad , \text{ for } j = 1,\ldots,(2^s+t-a_s)/2$$

and $$\bar{w}^s(j) = (2^s+t+a_s)/2+1-j \quad , \text{ for } j = (2^s+t-a_s)2+1,\ldots,(2^s+t+a_s)/2.$$

For example, $s = 3$, $t = 1$, and $a_s = 3$, $\bar{w}^3(j)$ are as follows.

j:	1	2	3	4	5	6
$\bar{w}^s(j)$:	4	4	4	3	2	1
	$\underbrace{\qquad\qquad}_{(2^s+t-a_s)/2}$			$\underbrace{\qquad\qquad}_{a_s}$		

Suppose we have found $\bar{w}^{i+1}(j)$, $j = 1,\ldots,m_{i+1}$. The weights $\bar{w}^i(j)$ are to be generated in the following way. Partition positions 1 to m_{i+1} into $(s-1)$ segments, each of length $q = m_{i+1}/(s-1)$; the first segment consists of the first q positions and the second segment consists of positions $q+1,\ldots,2q$ and so on. Let $X_{k,\ell}$ be the weight $\bar{w}^{i+1}(j)$ where $j = (k-1)q + \ell$, and $X_{k,\ell}$ be the weight $\bar{w}^i(j)$, where $j = (k-1)(q+a_i) + \ell$. From the set of the weights $\{X_{k,\ell} | 1 \le \ell \le q\}$, the set of weights $\{X_{k,\ell} | 1 \le \ell \le q+a_i\}$ is generated as two parts:

$$X_{k\ell} = X_{k\ell} \qquad \text{for } \ell = 1,\ldots,(q-a_i)/2$$

$$X_{k\ell} = \frac{(q+a_i)/2+1-\ell}{a_i+1} \cdot X_{k,(q-a_i)/2}, \text{ for } \ell = (q-a_i)/2+1,\ldots,(q+a_i)/2$$

and $$X_{k\ell} = X_{k,\ell-a_i} - X_{k,\ell-a_i} \quad , \text{ for } \ell = (q+a_i)/2+1,\ldots,(q+a_i)/2+a_i$$

$$X_{k\ell} = X_{k,\ell-a_i} \quad , \text{ for } \ell = (q+a_i)/2+a_i+1,\ldots,q+a_i \; .$$

For example, s = 4, t = 1 and $(a_1,a_2,a_3,a_4) = (0,1,1,9)$, the weights are generated as follows. (Notice that there is no duplication at the 1st boundaries, therefore $\bar{w}^2(j) = \bar{w}^1(j)$.)

$\bar{w}^4(j)$:	10	10	10	10	9	8	7	6	5	4	3	2	1
$\bar{w}^3(j)$:	10	10	10	10	9	8	4 | 3	6	5	4	3	2	1
$\bar{w}^2(j)$:	10	10	10	5 | 5	9	8	4	3	6	5	$\frac{5}{2}$ | $\frac{3}{2}$	2	1
$\bar{w}^1(j)$:	10	10	10	5	5	9	8	4	3	6	5	$\frac{5}{2}$ $\frac{3}{2}$	2	1
$\bar{w}(j)$:	$\frac{15}{2}$	10	10	5	5	9	8	4	3	6	5	$\frac{5}{2}$ $\frac{3}{2}$	2	1

The set of weights $\bar{w}^i(j)$ obtained at each iteration i and $w(k,\ell)$, for $1 \le k,\ell \le 2m_i$ given in Lemma 4.6 satisfies equation (4.2).

Theorem 4.2.

The procedure in Algorithm 4.1 correctly generates the weight distribution $\bar{w}(j)$, for $j \in I(2^s)$ for the extended array of $a(2^s+t) \times (2^s+t)$ array under the duplicating pattern $(a_1,\ldots,a_s)$.

Proof:

See Appendix A. □

Theorem 4.3.

Let A* be the extended array of a $(2^s+t) \times (2^s+t)$ array A under the duplicating pattern $(a_1,a_2,\ldots,a_s)$. The normalization factor $\sum_{i,j \in V} w(i,j)$, for a weight distribution w of a graph $G_A^* = (V,E)$, $V = \{(i,j) \mid i,j \in I(2^{s+1})\}$ is equal to $(2^s+t)(2^s+t-1)(a_s+1)^2$.

Proof:

Observe that

$$\sum_{\substack{V(i,j) \text{ contains} \\ \text{record } r}} w(i,j) = \begin{cases} \frac{1}{2}(a_s+1)^2 & \text{if } r \text{ is at the corner of } A \\ \frac{3}{4}(a_s+1)^2 & \text{if } r \text{ is at the lowest boundary} \\ (a_s+1)^2 & \text{otherwise.} \end{cases}$$

□

The limiting probability distribution p of the vertices of a graph G_A^* can be obtained by Lemmas 4.5 and 4.6; and Theorems 4.2 and 4.3.

Algorithm 4.1.

```
begin
    for j ∈ I(2^s) and i ∈ I(2^s), set dj and X^i(j) to zero;
    set a_0 to zero;
    for i ∈ I((2^s+t)/2), set w̄^s(i) to a_s+1;
    [ℓj is the number of distinct records between a j^th boundary
     and the nearest boundary which is higher than j^th]
    ℓ_{s+1} = n
    for j = 1 to s do ℓj = (ℓ_{j+1} + a_j)/2;
    j = s;
LP: while j ≥ L do
        begin
            for i = 1 to 2^{s-j} step 2 do
                begin
                    FPG(1 + (i-1)ℓj, j)
```

```
                    if j ≠ s then begin BPG(1+i•ℓ_j, j);
                                        d_j = d_j + 1;
                                  end;
               end;
          j = j-1;
     end;
for i = 2 to 2^s do      w̄(i) = w̄^L(i);
w̄(1) = w̄(1) • 3/4;
Procedure FWG (f,j):
     begin
          for i = f to f + ℓ_j - a_j - 1 do
               begin
                    w̄^j(i) = w̄^{j+1}(i - d_j a_j);
                    X^j(i) = X^{j+1}(i - d_j a_j);
               end;
          for i = f + ℓ_j - a_j to f + ℓ_j - 1 do
                    w̄^j(i) = (i_j+f-i)/(a_j+1) (w̄^j(f+ℓ_j-a_j-1) + X^i(f+ℓ_j-a_j-1))
     end;
```

```
Procedure BWG(f,j):
    begin
        for i = f to f + a_j - 1 do
            w̄^j(i) = w̄^{j+1}(i - (d_j+1)a_j) - w̄^i(i-a_j);
        for i = f + a_j to f + ℓ_j - 1 do
            begin
                w^j(i) = w̄^{j+1}(i - (d_j+1)a_j);
                X^j(i) = X^{j+1}(i - (d_j+1)a_j);
            end;
        X^i(f + a_j - 1) = w̄^j(f - 1);
    end:
end.
```

4.2.2. Calculation of average stretch

The average stretch of the arcs of a graph $G^*_A = (V,E)$ is defined as:

$$\sum_{e \in E} \Pr(e) \cdot \text{Stretch of } e.$$

It can be expressed as

$$\sum_{v \in V} \sum_{w \in \text{Succ}(v)} p(v) \cdot \Pr(w|v) \cdot \text{stretch of arc } (v,w)$$

Since the conditional probability $\Pr(w|v)$ is a constant $1/|\text{Succ}(v)|$ for all $w \in \text{Succ}(v)$, the average stretch is

$$\sum_{v \in V} p(v)/|\text{Succ}(v)| \cdot \sum_{w \in \text{Succ}(v)} \text{stretch of arc } (v,w). \qquad (4.3)$$

Let S(i,j) be the sum of the stretches of the arcs originating at a particular vertex w which is at the horizontal i^{th} and the vertical j^{th} boundaries, that is

$$S(i,j) = \sum_{w \in Succ(v)} \text{stretch of } (v,w)$$

In the following lemma, it will be shown that S(i,j) is a function of i and j.

Lemma 4.7.

$$S(i,j) = \begin{cases} 12 + 4i + 4j & \text{for } i,j > 0 \\ 8 + 4j & \text{for } i = 0 \text{ and } j > 0 \\ 10 + 4i & \text{for } i > 0 \text{ and } j = 0 \\ 6 & \text{for } i = j = 0 \end{cases}$$

Proof:

In Figure 11 it is observed that there is a horizontal arc and a vertical arc directed from a vertex v at the horizontal i^{th} boundary and at the vertical j^{th} boundary, and these boundaries are the highest ones which these arcs must cross. If j (i) is greater than zero, there is an additional vertical (horizontal) arc directed from v and the horizontal (vertical) 0^{th} boundary is the highest boundary that this arc crosses. By Lemma 4.1, the lemma follows.

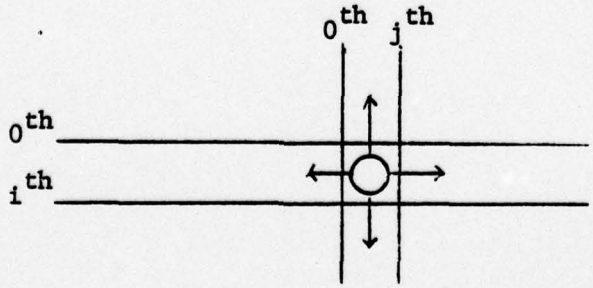


Figure 11. Arcs directed from a vertex at the horizontal i^{th} and the vertical j^{th} boundaries.

By Theorem 4.3 and equation (4.3), and the Lemmas 4.5 and 4.6, the average stretch ES of a graph G_A^* where A* is the extended array of a $(2^s+t) \times (2^s+t)$ array under the duplicating pattern $(a_1,a_2,\ldots,a_s)$ can be expressed as follows.

$$ES = \frac{4}{(2^s+t)(2^s+t-1)(a_s+1)^2}\left(\frac{1}{2}\cdot\frac{(a_s+1)^2}{2}\, s(0,0) + \frac{1}{3}\cdot\frac{3}{4}(a_s+1)F + \frac{1}{3}\cdot\frac{3}{4}(a_s+1)H + \frac{1}{4}G\right) \tag{4.4}$$

where $F = \sum_{i \in I(s)} W_i\, S(i,0)$,

$H = \sum_{i \in I(s)} W_i\, S(0,i)$,

$G = \sum_{i \in I(s)} W_i \sum_{j \in I(s)} W_j\, S(i,j)$

and W_i = the cumulative weights of the vertices at the vertical i^{th} boundaries of the row 2^s

$$= \sum_{j \in I(2^s-1-i)} (\overline{w}(2^i+(j-1)2^{i+1}) + \overline{w}(2^i+(j-1)2^{i+1}+1)), \quad \text{for } i \in I(s-1)$$

$W_s = 1$.

We are going to use the following lemma on the inequality of sums of two sequences of real numbers to show how the average stretch depends on the weight distribution derived from some duplicating pattern $(a_1,\ldots,a_s)$.

Lemma 4.7.

For two sequences of real numbers $(b_1,\ldots,b_s)$ and $(c_1,\ldots,c_s)$,

If $\sum_i i c_i < \sum_i i b_i$ and $\sum_i c_i = \sum_i b_i$

then $\sum_i c_i \sum_j c_j (i+j) < \sum_i b_i \sum_j b_j (i+j)$

Proof:

$$\sum_i c_i \sum_j c_j (i+j) = \sum_i i c_i \sum_j c_j + \sum_i c_i \sum_j j c_j$$

$$= \sum_i i c_i \sum_j b_j + \sum_i b_i \sum_j j c_j$$

$$< \sum_i i b_i \sum_j b_j + \sum_i b_i \sum_j j b_j$$

$$= \sum_i b_i \sum_j b_j (i+j). \qquad \square$$

Theorem 4.4.

If $\sum_i i \frac{W'_i}{a'_s+1} < \sum_i i \frac{W''_i}{a''_s+1}$, where W'_i and W''_i are the cumulative weights at the i^{th} boundaries of $G_{(A^*)'}$ and $G_{(A^*)''}$ respectively, $(A^*)'$ and $(A^*)''$ are extended arrays under $(a'_1,\ldots,a'_s)$ and $(a''_1,\ldots,a''_s)$ respectively, then $ES' < ES''$.

Proof:

$$ES = \frac{1}{n(n-1)}\left[6 + \sum_i \frac{W_i}{a_s+1}(18+8i) + \sum_i \frac{W_i}{a_s+1} \sum \frac{W_j}{a_s+1}(12+4i+4j)\right]$$

$$ES' - ES'' = \frac{1}{n(n-1)}\left(8\left(\sum_i \frac{W'_i}{a_s+1} i - \sum_i \frac{W''}{a_s+1} i\right) + 18\left(\sum_i \frac{W'_i}{a_s+1} - \sum_i \frac{W''_i}{a_s+1}\right)\right.$$

$$+ 12\left(\sum_i \frac{W'_i}{a'_s+1} \sum_j \frac{W'_j}{a'_s+1} - \sum_i \frac{W''_i}{a''_s+1} \sum_j \frac{W''_j}{a''_s+1}\right)$$

$$\left. + 4\left(\sum_i \frac{W'_i}{a'_s+1} \sum_j \frac{W'_j}{a'_s+1} (i+j) - \sum_i \frac{W''_i}{a''_s+1} \sum_j \frac{W''_j}{a''_s+1}(i+j)\right)\right)$$

we can show that $\sum_i \frac{W'_i}{a'_s+1} = \sum_i \frac{W''_i}{a''_s+1} = \frac{n}{2} - 1$ and by the hypothesis, $\sum_i i \frac{W'_i}{a'_s+1} < \sum_i i \frac{W''_i}{a''_s+1}$. Therefore, it follows from Lemma 4.3 that $ES' - ES'' < 0$. $\square$

Since the set of arcs in a graph with no duplication is uniformly distributed, the average stretch ES is the total stretch TS of all arcs divided by the number of arcs. The total stretch TS can be expressed as

$$TS = \Sigma\, m_{ij} S(i,j) \quad \text{where } m_{ij} = \text{number of vertices at the horizontal } i^{th} \text{ boundary and the vertical } j^{th} \text{ boundary.} \qquad (4.5)$$

CHAPTER 5

DUPLICATING PATTERNS

5.1. Feasible Duplicating Patterns

It has been mentioned briefly that a duplicating pattern is a sequence of nonnegative integers indicating exactly the number of duplications at each boundary. It is obvious that the duplicating pattern cannot be any arbitrary sequence by observing that there are constraints both on the amount of duplication and the number of boundaries. A feasible duplicating pattern is defined precisely as follows:

Let A* be the extended array of an $n \times n$ array, where $n = 2^s + t$, for some $t \in I(2^s-1)$, under the duplicating pattern $(a_1,a_2,\ldots,a_s)$. Then $(a_1,a_2,\ldots,a_s)$ is a feasible duplicating pattern if a_i's satisfy the equation

$$\sum_{i=1}^{s} a_i 2^{s-i} = 2^s - t \qquad (5.1)$$

where 2^{s-i} is the number of vertical i^{th} boundary in a row and $2^s - t$ is the maximum number of duplications in a row.

We are only interested in feasible duplicating patterns; thus, for brevity, a duplicating pattern means a feasible duplicating pattern. In the remaining sections of this chapter, the effect of duplicating patterns on the average stretch will be investigated.

5.2. Some Special Duplicating Patterns

5.2.1. No duplication

For a graph $G_A = (V,E)$ of an $n \times n$ array A with no duplication, all the arcs in E are equally probable (refer to Corollary 4.1). Therefore, the average stretch will be the total stretch divided by the cardinality of E. When n is a power of 2, let $n = 2^s$. The total stretch TS is

defined by the following recurrence:

$$TS(1) = 0$$

$$TS(n) = 4\,TS(n/2) + 2n(2\log((n/2)^2) + 2) + 2n(2\log((n/2)^2) + 4),$$

where the first term on the right hand side of the equation is the total stretch of arcs within each of the four $n/2 \times n/2$ subarrays (refer to Figure 12), the second term is the stretch of all the arcs crossing the vertical s^{th} boundary, and the last term is the stretch of all arcs crossing the horizontal s^{th} boundary.

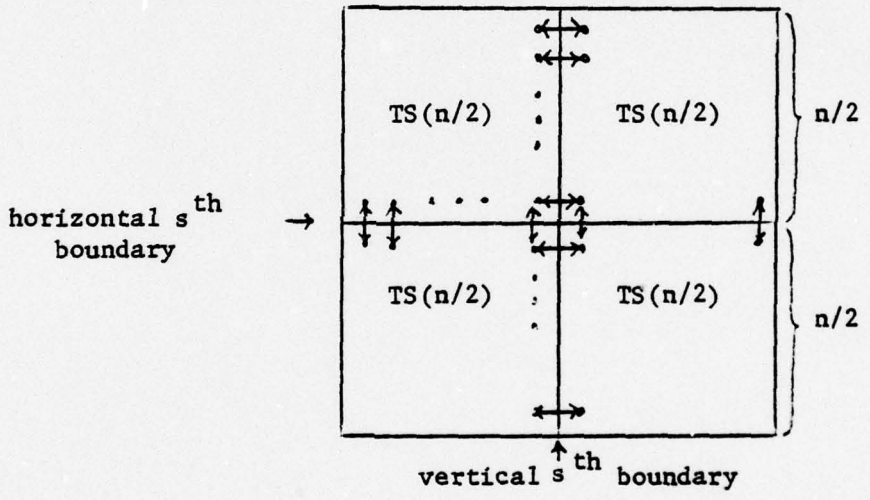


Figure 12. Illustration of the recurrence of the total stretch

The solution to this recurrence equation is

$$TS(n) = 28n^2 - 16\,n\log n - 28n$$

There are $4n(n-1)$ edges in E, thus the average stretch ES is

$$ES = \frac{28n^2 - 16\,n\log n - 28n}{4n^2 - 4n},$$

which is asymptotically equal to 7 as n approaches infinity.

When $n > 2^s$ is not a power of 2, there is no simple recurrence for solving the total stretch TS since the storage mapping is not symmetrical about the s^{th} boundaries of the array. Equation (4.5) has to be used and $m_{i,j}$, the number of vertices at a horizontal i^{th} boundary and a vertical j^{th} boundary, is given below

$$m_{i,j} = X_i X_j \qquad \text{for } i,j \in I(s),\ n = 2^s + t,\ 1 \le t \le 2^s$$

$$m_{0,0} = 1$$

$$X_i = 2^{s-i} + 2\left\lceil \frac{t - 2^i + 1}{2^{i+1}} \right\rceil + y_i \qquad \text{for } i \in I(s-1)$$

$$\text{where } y_i = \begin{cases} -1 & \text{if } \left\lceil \frac{t - 2^i + 1}{2^{i+1}} \right\rceil (2+2^{i+1}) - 2^{i+1} > t - 2^i + 1 \\ 0 & \text{otherwise} \end{cases}$$

and $X_s = 2$

$X_0 = 1$

Thus, $ES = \dfrac{\sum_{i=0}^{s} \sum_{j=0}^{s} m_{i,j}\, s(i,j) - n*(8k-6)}{4n(n-1)}$ where $k = k_\ell$ if $t = 2^{k_1} + 2^{k_2} + \ldots + 2^{k_\ell}$

$k_1 > k_2 > \ldots > k_\ell$. Instead of simplifying the expression on the right hand side, which requires some tedious algebraic manipulations, we evaluate ES for $n = 2^s + 1$, $2^s + 2^{s-1}$, $2^s + 2^s - 1$. The numerical results are shown in Table 1. From these results, we can say that for these particular $n = 2^s + 1$, $2^s + 2^{s-1}$, $2^s + 2^s - 1$, ES is asymptotic to 7. Since these values of n are the two extremes and the median of the interval comprised between two consecutive powers of 2, it is resonable to say that ES is asymptotically equal to 7, for all integer n. This value will be used as reference to measure the improvement of ES when duplication is introduced. In the following subsections, three special duplicating patterns will be investigated.

s	$ES(2^s+1)$	$ES(2^s+2^{s-1})$	$ES(2^{s+1}-1)$
2	6.00000000	5.39999997	5.66666668
3	6.50000000	5.90909093	6.14285713
4	6.75000000	6.30434781	6.46666663
5	6.87500000	6.57446807	6.67741936
6	6.93750000	6.74736839	6.80952382
7	6.96875000	6.85340315	6.88976377
8	6.98437500	6.91644906	6.93725490
9	6.99218750	6.95306390	6.96477496
10	6.99609375	6.97394138	6.98044967
11	6.99804687	6.98567241	6.98925262
12	6.99902349	6.99218624	6.99413913
13	6.99951177	6.99576795	6.99682581
14	6.99975591	6.99772131	6.99829089
15	6.99987792	6.99877929	6.99908447
16	6.99993896	6.99934899	6.99951177
17	6.99996942	6.99965417	6.99974071
18	6.99998474	6.99981695	6.99986273
19	6.99999237	6.99990344	6.99992758
20	6.99999624	6.99994921	6.99996191
21	6.99999803	6.99997335	6.99998009
22	6.99999904	6.99998611	6.99998950
23	6.99999952	6.99999278	6.99999457
24	6.99999976	6.99999624	6.99999713
25	6.99999988	6.99999803	6.99999856
26	7.00000000	6.99999916	6.99999940
27	6.99999994	6.99999958	6.99999964
28	7.00000005	6.99999976	6.99999982
29	7.00000005	7.00000000	6.99999994
30	7.00000005	7.00000000	7.00000005
31	7.00000011	7.00000005	7.00000005
32	7.00000005	7.00000005	7.00000005
33	7.00000005	7.00000005	7.00000005

Table 1. Average stretch with no duplication.

5.2.2. $(0,0,\ldots,2^s-t)$

When n is not a power of 2, there are many ways to extend an $n \times n$ array to a $2^{\lceil \log n \rceil} \times 2^{\lceil \log n \rceil}$ array. Arcs whose images cross a high boundary correspond to a large stretch under the storage mapping, which contributes heavily to the average stretch. Thus, we would want to avoid using arcs which cross the highest boundaries. This can be done by placing duplicates of those records, which these arcs are trying to reach across the highest boundaries, in the same subarrays from which these arcs are directed. The first duplicating pattern $(0,0,\ldots,0,2^s-t)$ that we investigate for an $(2^s+t) \times (2^s+t)$ array is doing exactly what we have just described.

Algorithm 4.1 can be used to generate the weight distribution and then equation (4.4) is applied to obtain the average stretch ES(n). For the pattern $(0,0,\ldots,2^s-t)$, the algorithm 4.1 gives the following weight distribution

$i =$	1	2	. . .	$t+1$	$t+2$	...	2^s
$\bar{w}(i) =$	$\frac{3}{4}(2^s-t+1)$	2^s-t+1		2^s-t+1	2^s-t		1

For $t = 1, 2^{s-1}, 2^s-1$, closed form expressions for the W_i's used in equation (4.4) can be easily obtained from the weight distribution (refer to Appendix B). Numerical results are used to study the behavior of ES_{A^*}. The improvement in ES_{A^*} with duplication can be measured in terms of the figure of merit $\frac{ES_A - ES_{A^*}}{ES_A} \times 100\%$. Table 2 shows the average stretch

s	ES(2^s+1)	I%	ES(2^s+2^{s-1})	I%	ES(2^{s+1}-1)	I%
2	4.86666673	20.8	4.75000000	9.9	5.00000000	11.8
3	5.61818188	16.3	5.43750000	4.9	5.71428573	7.0
4	6.14975852	11.3	5.98437500	2.5	6.19999999	4.1
5	6.49436802	7.3	6.37109375	1.2	6.51612902	2.4
6	6.70653915	4.5	6.62402344	0.6	6.71428573	1.4
7	6.83278298	2.7	6.78100592	0.3	6.83464569	0.8
8	6.90608621	1.5	6.87493902	0.2	6.90588236	0.5
9	6.94786906	0.9	6.92967224	0.1	6.94716245	0.3
10	6.97134060	0.5	6.96093369	0.0	6.97067452	0.1
11	6.98437119	0.3	6.97851467	0.0	6.98387885	0.1
12	6.99153543	0.2	6.98828101	0.0	6.99120879	0.0
13	6.99544245	0.1	6.99365240	0.0	6.99523866	0.0
14	6.99755859	0.0	6.99658203	0.0	6.99743640	0.0
15	6.99869794	0.0	6.99816901	0.0	6.99862671	0.0
16	6.99930829	0.0	6.99902350	0.0	6.99926764	0.0
17	6.99963385	0.0	6.99948126	0.0	6.99961096	0.0
18	6.99980676	0.0	6.99972540	0.0	6.99979407	0.0
19	6.99989831	0.0	6.99985504	0.0	6.99989134	0.0
20	6.99994665	0.0	6.99992377	0.0	6.99994284	0.0
21	6.99997211	0.0	6.99996001	0.0	6.99997002	0.0
22	6.99998546	0.0	6.99997902	0.0	6.99998432	0.0
23	6.99999237	0.0	6.99998909	0.0	6.99999183	0.0
24	6.99999607	0.0	6.99999434	0.0	6.99999565	0.0
25	6.99999785	0.0	6.99999708	0.0	6.99999780	0.0
26	6.99999899	0.0	6.99999851	0.0	6.99999899	0.0
27	6.99999964	0.0	6.99999929	0.0	6.99999940	0.0
28	6.99999982	0.0	6.99999970	0.0	6.99999976	0.0
29	6.99999994	0.0	6.99999988	0.0	6.99999994	0.0
30	7.00000006	0.0	6.99999994	0.0	6.99999994	0.0
31	7.00000018	0.0	7.00000006	0.0	7.00000006	0.0
32	7.00000006	0.0	7.00000006	0.0	7.00000006	0.0
33	7.00000006	0.0	7.00000006	0.0	7.00000006	0.0
34	7.00000006	0.0	7.00000006	0.0	7.00000006	0.0

Table 2. Average stretch and improvement I under $(0,0,\ldots,2^s-t)$

ES_{A*} (n) and their improvements for $n = 2^s+1, 2^s+2^{s-1}, 2^s+2^s-1, s \in \{2,3,\ldots,34\}$. The improvements diminish rapidly to zero as n increases. Therefore $(0,0,\ldots,2^s-t)$ is certainly not a good duplicating pattern.

There is nothing that can be done for $n = 2^s + 2^s - 1$, since $(0,0,\ldots,0,1)$ is the only feasible duplicating pattern. There is more freedom of choice of duplicating patterns for $n < 2^{s+1} - 1$, and this is maximum for $n = 2^s + 1$. Therefore, the remaining two subsections will be dedicated to the study of the case $n = 2^s + 1$, in order to learn the effect of the choice of a_i's on the average stretch.

5.2.3. $\underline{(0,1,1,\ldots,1,2^{s-1}+1)}$

We have seen that duplicating only at the highest boundaries is not a good strategy. In this section, a more uniform distribution of a_i's, namely $(0,1,1,\ldots,1,2^{s-1}+1)$, for $n = 2^s + 1$, is investigated. This duplicating pattern allows one duplication near each boundary except the lowest and the highest boundaries. It allows no duplication near the lowest boundaries and gives whatever number of duplications remained to the highest boundaries.

Observing the weight distribution $\bar{w}(i)$'s, closed form expression of W_i's in terms of n and i are obtained after some long algebraic manipulation; they are shown in Appendix B. These W_i's are substituted into equation (4.4) to calculate the average stretch ES_{A*}. The results show that for a large value of n, ES_{A*} is approximately 6.33 (and this appears to be the asymptotic value) which produces an improvement of about 9.5%. When n is smaller, the improvement is larger (Table 3).

s	$ES(2^s+1)$	I%
2	4.75000000	20.8
3	5.24999994	19.2
4	5.65000010	16.3
5	5.91666669	13.9
6	6.09007365	12.2
7	6.19460231	11.1
8	6.25576931	10.4
9	6.29051590	10.0
10	6.30993891	9.8
11	6.32064945	9.7
12	6.32650077	9.6
13	6.32967216	9.6
14	6.33138073	9.5
15	6.33229584	9.5
16	6.33278424	9.5
17	6.33304334	9.5
18	6.33328079	9.5
19	6.33325332	9.5
20	6.33329147	9.5
21	6.33331138	9.5
22	6.33332211	9.5
23	6.33332735	9.5
24	6.33333033	9.5
25	6.33333170	9.5
26	6.33333254	9.5
27	6.33333278	9.5
28	6.33333319	9.5
29	6.33333325	9.5
30	6.33333325	9.5
31	6.33333325	9.5
32	6.33333325	9.5
33	6.33333325	9.5

Table 3. Average stretch and improvement I under $(0,1,1,\ldots,1,2^{s-1}+1)$.

For $n = 2^s + 1$, $(0,1,1,\ldots,1,2^{s-1}+1)$ is superior to $(0,0,\ldots,0,2^s-1)$. For arbitrary $2^s + 1 < n < 2^{s+1} - 1$, it is also expected that duplication at lower boundaries is desirable: indeed, the number of low boundaries is very large, which corresponds to a high frequency of low-boundary crossing in the random-walk hypothesis we have made.

5.3.4. <u>(0,1,2,4,4,...,4,7)</u>

We have seen that $(0,1,1,\ldots,1,2^{s-1}+1)$ is a better duplicating pattern than $(0,0,\ldots,0,2^s-1)$ for $n = 2^s + 1$. Are there patterns which are still better? In this section, this question is answered by showing that $(0,1,2,4,4,\ldots,4,7)$, for $n = 2^s + 1$, $s = 5,6,\ldots$, is better.

Algorithm 4.1 is used to generate the weight distribution $\bar{w}(i)$'s for $s = 5$ and the W_i's used in equation (4.4) are calculated. Since a_i's, $i = 4,5,\ldots,s-1$, have the same value, W_i's for $s = 5$ is easily extended to those for $s > 5$. The closed form expressions for these W_i's are given in Appendix B.

Table 4 shows that for large values of n, ES_{A*} approaches 6.13 which gives an improvement of 12.4%. This duplicating pattern is a combination of the heuristic criteria presented in the previous two sections, namely (i) crossing of high boundaries contributes heavily to the average stretch and (ii) low boundaries are frequently crossed. Therefore, this pattern suggests that there are duplications at higher boundaries and at the same time there are duplications at lower boundaries.

s	$ES(2^s+1)$	I%
5	5.8395834	15.1
6	5.9692709	14.0
7	6.0427083	13.3
8	6.0837240	12.9
9	6.1063802	12.7
10	6.1187826	12.5
11	6.1255209	12.5
12	6.1291586	12.4
13	6.1311117	12.4
14	6.1321554	12.4
15	6.1327108	12.4
16	6.1330052	12.4
17	6.1331608	12.4
18	6.1332430	12.4
19	6.1332861	12.4
20	6.1333085	12.4
21	6.1333205	12.4
22	6.1333266	12.4
23	6.1333300	12.4
24	6.1333316	12.4
25	6.1333325	12.4
26	6.1333328	12.4
27	6.1333329	12.4
28	6.1333333	12.4
29	6.1333333	12.4
30	6.1333333	12.4
31	6.1333333	12.4
32	6.1333333	12.4
33	6.1333333	12.4
34	6.1333333	12.4

Table 4. Average stretch and improvement I under (0,1,2,4,4,...,4,7).

This pattern seems to be quite good. Is it optimal? For arbitrary n, what is a good duplicating pattern? These two questions will be examined in the remaining section of this chapter.

5.3. Optimal Duplicating Patterns

An optimal duplicating pattern for an $n \times n$ array A is one under which A is extended to $\bar{A}^*$ and $ES_{\bar{A}^*}$ is smallest among all ES_{A^*}, where A^* is an extended array of A under some duplicating pattern.

An optimal duplicating pattern can be found by exhaustive search. The algorithn is given as follows.

Algorithm 5.1.

Input: An $n \times n$ array A, where $n = 2^s + t$

Output: An optimal duplicating pattern for A.

Method:

1. Call the procedure PG (s,n) to generate the set DP of all feasible duplicating patterns for A.
2. For each pattern in DP, use algorithm 4.1 to obtain the probability distribution of the set of vertices in the graph of the extended array A^* of A. Then apply equation (4.4) to obtain the average stretch ES_{A^*}.
3. Output a duplicating pattern $d \in DP$, such that $\bar{A}^*$ is the extended array of A under d and $ES_{\bar{A}^*} < ES_{A^*}$.

```
Procedure  PG(s,t)
     DP ← φ;
     a_s ← 2^s-t; a_i ← 0; i = 1,2,...,s-1
     j ← s-2;
     while j > 0  do
          begin
               j ← s-2;
               while a_s ≥ 0  do
                    begin
                         add (a_1,a_2,...,a_s) to DP;
                         a_{s-1} ← a_{s-1} + 1;
                         a_s ← a_s - 2;
                    end;
               repeat
                    if j ≠ 0 then a_j ← a_j + 1
                    for i ← j+1 until s-1 do a_i ← 0;
                    a_s ← 2^s - t - Σ_{i=1}^{j} 2^{s-1} a_i
                    if a_s < 0 then j ← j - 1
               until a_s ≥ 0;
          end;
```

5.4. Near Optimal Duplicating Patterns for $n = 2^s + 1$

For large value of s and small value of t, there are so many duplicating patterns that it becomes impractical to obtain an optimal duplicating pattern for each n by algorithm 5.1. It is preferable to have a general duplicating scheme which guarantees small average stretch

for every n. The two heuristics used to construct an efficient duplicating scheme are as follows:

1) An array composed of smaller arrays with good structure has a good structure;

2) If W_i's are small on the lower end and large on the higher end then $\sum_{i=1}^{s} i \frac{W_i}{c}$ is small, which implies small average stretch.

These will be explained and justified in more detail in this section.

Theorem 5.1.

If A is an $(2^s+1) \times (2^s+1)$ array, s is an integer greater than or equal to 6 and the duplicating pattern is $(0,1,2,4,\ldots,4,a_{s-1},a_s)$ then the minimum value of ES_{A*} is achieved when

$$(a_{s-1},a_s) = (4,7) \quad \text{if} \quad 6 \le s \le 9$$

$$(a_{s-1},a_s) = (5,5) \quad \text{if} \quad s \ge 10.$$

Proof: (by exhausting all the possible combinations of (a_{s-1},a_s))

Since $\sum_{i=1}^{s} a_i \, 2^{s-i} = 2^s - 1$ and $(a_1,a_2,\ldots,a_{s-2}) = (0,1,2,4,\ldots,4)$, $2a_{s-1} + a_s$ must be equal to 15. Thus, there are 8 possible pairs of (a_{s-1},a_s). They are $(i,15-2i)$ for $i = 0,1,\ldots,7$. Recall that the W_i's are the cumulative weights at i^{th} boundaries. For each (a_{s-1},a_s) compute the difference D_j, $j = 0,1,\ldots,7$

$$D_j = \sum_{i=1}^{s} i \frac{W_i^{(4)}}{5} - \sum_{i=1}^{s} i \frac{W_i^{(j)}}{a_s+1}$$ where $W_i^{(j)}$ are the cumulative weights when when $(a_{s-1},a_s) = (j,15-2j)$. The closed form expressions for these $W_i^{(j)}$ shown in the appendix B are used in computing D_j. The D_j are:

$$D_j = \begin{cases} 277/60 - 123/80 \cdot s & \text{for } j = 0 \\ 89/28 - 433/280 \cdot s & \text{for } j = 1 \\ 313/360 - 27/120 \cdot s & \text{for } j = 2 \\ 23/120 - 3/40 \cdot s & \text{for } j = 3 \\ -29/120 + 3/120 \cdot s & \text{for } j = 5 \\ -151/280 - 3/280 \cdot s & \text{for } j = 6 \\ 17/40 - 9/40 \cdot s & \text{for } j = 7 \end{cases}$$

For $s \geq 6$, D_j is negative when $j = 0,1,\ldots,3,6,7$ when $j = 5$, D_j is negative for $6 \leq s \leq 9$ and D_j is positive when $s \geq 10$. Hence, the theorem follows from Theorem 4.5. □

Since D_5 is $0(\log(n-1))$, the difference in the average stretch between the patterns $(0,1,2,4,\ldots,4,7)$ and $(0,1,2,4,\ldots,5,5)$ is $0(1/n)$ which approaches zero when n is large. Numerical results show that for $s \geq 24$, the average stretches for both patterns are the same.

Using algorithm 5.1, the optimal duplicating patterns for $n = 2^s + 1$ and $s = 3,4,5,6$ are found to be $(0,1,5)$, $(0,1,2,7)$, $(0,1,2,4,7)$ and $(0,1,2,4,4,7)$ respectively. Thus, the duplicating pattern $(0,1,2,4,\ldots,4,7)$ is inductively an efficient pattern. With the evidence of Theorem 5.1, a reasonable conjecture is that $(0,1,2,4,\ldots,4,7)$ is near optimal for $n = 2^s + 1$.

When n is not equal to $2^s + 1$, $(0,1,2,4,\ldots,4,7)$ certainly cannot be an optimal duplicating pattern since it is not a feasible duplicating pattern. Further research is needed in order to obtain some near optimal duplicating schemes for $n \neq 2^s + 1$.

CHAPTER 6

CONCLUSION

Any scheme that stores arrays in a linear memory induces unbounded average loss of proximity. DeMillo, Eisenstat and Lipton [4] have shown that when square arrays are stored in a binary tree memory structure, the average loss of proximity is bounded by a constant 7. In this paper, duplication of items of an array is introduced to decrease the average loss of proximity in arrays when they are stored in a binary tree memory structure. It is shown that some duplication can induce as much as a 12% decrease in average loss of proximity. Moreover, the duplication ratio is limited in these schemes, since we do not use a deeper binary tree memory structure BTMS than is needed when no duplication is used. Therefore, the duplicating patterns defined in this paper yield a very effective space-time tradeoff.

In Chapter 5, it is shown that duplication at low boundaries is as important as at high boundaries in preserving the average loss of proximity in arrays. And the distribution of the number of duplications at the various boundaries must be carefully chosen in order to achieve decrease in the average loss of proximity.

REFERENCES

1. A. Aho, J. Hopcroft and J. D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, 1974.

2. A. L. Rosenberg, "Preserving Proximity in Arrays," SIAM Journal of Computing, Vol. 4, No. 4, December 1975.

3. D. Bollman, "On Preserving Proximity in Extendible Arrays," SIAM Journal of Computing, Vol. 5, No. 2, June 1976.

4. R. A. DeMillo, S. C. Eisenstat and R. J. Lipton, "Preserving Average Proximity in Arrays," School of Information and Computer Science, Georgia Institute of Technology, Atlanta, Georgia, September 1976.

5. R. G. Gallager, Information Theory and Reliable Communication Jony Wiley and Sons, New York, 1968.

APPENDIX A

The proof of Theorem 4.2 is given as follows.

By backward induction on L

Basis: L = S

FWG(1,s) generates $\bar{w}^s(i) = a_s + 1$ for $i \in I(\ell_s - a_s)$

and $\bar{w}^s(i) = \ell_s + 1 - i$ for $i \in I(\ell_s) - I(\ell_s - a_s)$

Since $a_s \geq 1$, $\bar{w}^s(\ell_s - 1) = 2$

By Lemma 4.6, $w(\ell_s - 1, j)$ is set to $\bar{w}^s(\ell_s - 1)\bar{w}^s(j) = 2\bar{w}^s(j)$

$j \in I(\ell_s)$

Since $\bar{w}(1)$ is set to $3/4(a_s+1)$ and by Lemma 4.6,

$w(1,1)$ is set to $1/2(a_s+1)^2$

Now, we show that $\bar{w}(i)$, for all $i \in I(\ell_s)$, satisfies

equation (4.3)

$$\bar{w}(1) = 2 \cdot \bar{w}(1) \cdot 1/3 + \bar{w}(2) \cdot 1/4$$

$$= 3/4(a_s+1)$$

$$\bar{w}(2) = \begin{cases} 2\bar{w}(2) \cdot 1/4 + \bar{w}(1) \cdot 1/3 + \bar{w}(3) \cdot 1/4 & \text{if } 2 < \ell_s - a_s \\ 2\bar{w}(2) \cdot 1/4 + \bar{w}(1) \cdot 1/3 + \bar{w}(3) \cdot 1/4 + \bar{w}(\ell_s) 1/4 & \text{if } 2 = \ell_s - a_s \\ \bar{w}(1) \cdot 1/3 + \bar{w}(1) \cdot 1/3 & \text{if } 2 > \ell_s - a_s \end{cases}$$

$$= \begin{cases} 1/2(a_s+1) + 1/4(a_s+1) + 1/4(a_s+1) \\ 1/2(a_s+1) + 1/4(a_s+1) + 1/4\, a_s + 1/4 \\ 1/2(a_s+1) \end{cases}$$

$$= \begin{cases} a_s + 1 & \text{if } 2 \leq \ell_s - a_s \\ \ell_s + 1 - 2 & \text{if } 2 > \ell_s - a_s \end{cases}$$

$$\begin{aligned}
\bar{w}(i) &= 2\bar{w}(i)\cdot 1/4 + \bar{w}(i-1)\cdot 1/4 + \bar{w}(i+1)\cdot 1/4, \quad i \in I(\ell_s - a_s - 1) - I(2) \\
&= 1/2(a_s+1) + 1/4(a_s+1) + 1/4(a_s+1) \\
&= a_s + 1 \\
\bar{w}(i) &= 2\bar{w}(i)\cdot 1/4 + \bar{w}(i-1)\cdot 1/4 + \bar{w}(i+1) 1/4 + \bar{w}(\ell_s)\cdot 1/4, \quad i = \ell_s - a_s \neq 2 \\
&= 1/2(a_s+1) + 1/4(a_s+1) + 1/4 \cdot a_s + 1/4 \\
&= a_s + 1 \\
\bar{w}(i) &= 2\bar{w}(i)\cdot 1/4 + \bar{w}(i-1)\cdot 1/4 + \bar{w}(i+1)\cdot 1/4, \quad i \in I(\ell_s - 1) - I(\ell_s - a_s) \\
&= 1/2(\ell_s + 1 - i) + 1/4(\ell_s + 1 - i + 1) + 1/4(\ell_s + 1 - i - 1) \\
&= \ell_s + 1 - i \\
\bar{w}(i) &= 2\bar{w}(i)\cdot 1/4 + \bar{w}(i-1)\cdot 1/4 \qquad\qquad i = \ell_s \\
&= 1/2 + 1/2 \\
&= 1
\end{aligned}$$

Assume that L = k, the algorithm correctly generates the weight distribution $\bar{w}^k(i)$, $i \in I(2^{s-k}\ell_k)$, that is $\bar{w}^k(i)$ satisfies equation (4.2) for $i \in I(2^{s-k}\ell_k)$.

Suppose $a_{k-1} > 0$, that is L = k-1, the loop LP in algorithm 4.1 has to be executed once more after $\bar{w}^k(i)$, $i \in I(2^{s-k}\ell_k)$, are found.

Each BWG(f,k) generates the weights of a_k cells which contain duplicate copies of some records and their copies the next $\ell_k - a_k$ weights from $\bar{w}^{k+1}$'s. Thus, each BWG will shift the records a_k places to the right. This explains the term $d_k \cdot a_k$ in the parameter of $\bar{w}^{k+1}$ in procedures FWG and BWG.

At each $(k-1)^{th}$ boundary, a_{k-1} records are duplicated and each of the 2^{s-k} sections of length ℓ_k is divided into 2 sections. Therefore, there are $2^{s-(k-1)}$ sections, each of length $(\ell_k + a_{k-1})/2$ when the loop LP is executed the $(s + 1-(k-1))^{th}$ time. FWG and BWG are alternately applied to each section.

FWG$(1+(i-1)\ell_{k-1},k-1)$ generates $\overline{w}^{k-1}(j), j \in I(i\cdot\ell_{k-1})-I((i-1)\ell_{k-1})$, $i \in \{1,3,\ldots,2^{s-k+1}-1\}$

For $j \in I(i\cdot\ell_{k-1}-a_{k-1}-1) - I((i-1)\cdot\ell_{k-1})$

$$\overline{w}^{k-1}(j) = \overline{w}^{k-1}(j-1)\cdot 1/4 + \overline{w}^{k-1}(j+1)\cdot 1/4 + 2\overline{w}^{k-1}(j)\cdot 1/4$$
$$= 1/4(\overline{w}^{k}(j-1-d_{k-1}a_{k-1}) + \overline{w}^{k}(j+1-d_{k-1}a_{k-1}) + 2\overline{w}^{k}(j-d_{k-1}a_{k-1})$$
$$= 1/4(4\overline{w}^{k}(j-d_{k-1}a_{k-1}))$$

For $j = i\cdot\ell_{k-1} - a_{k-1}$

$$\overline{w}^{k-1}(j) = \overline{w}^{k-1}(j-1)\cdot 1/4 + \overline{w}^{k-1}(j+1)\cdot 1/4 + \overline{w}^{k-1}(i\cdot\ell_{k-1}+1)\cdot 1/4$$
$$+ 2\overline{w}^{k-1}(j)\cdot 1/4$$
$$= 1/4[\overline{w}^{k}(j-1-d_{k-1}a_{k-1} + \overline{w}^{k-1}(j+1) +$$
$$\overline{w}^{k}(i\cdot\ell_{k-1}+1-(d_{k-1}+1)a_{k-1}) - \overline{w}^{k-1}(i\cdot\ell_{k-1}-a_{k-1})$$
$$+ 2\overline{w}^{k}(j-d_{k-1}a_{k-1})]$$
$$= \overline{w}^{k}(j-d_{k-1}a_{k-1})$$

For $j \in I(i\cdot\ell_{k-1}-1) - I(i\cdot\ell_{k-1}-a_{k-1})$

$$\overline{w}^{k-1}(j) = 1/4(\overline{w}^{k-1}(j-1)+\overline{w}^{k-1}(j+1)+2\overline{w}^{k-1}(j))$$
$$= 1/4\cdot 1/(a_{k-1}+1)\cdot(\overline{w}^{k-1}(i\cdot\ell_{k-1}-a_{k-1}) + X^{k-1}(i\cdot\ell_{k-1}-a_{k-1}))$$
$$(i\cdot\ell_{k-1}+1-(j-1)+i\cdot\ell_{k-1}+1-(j+1)+2i\cdot\ell_{k-1}+1-j)$$
$$= 1/(a_{k-1}+1)\cdot(\overline{w}^{k-1}(i\cdot\ell_{k-1}-a_{k-1}) + X^{k-1}(i\cdot\ell_{k-1}- a_{k-1}))$$
$$(i\cdot\ell_{k-1}+1-j)$$

For $j = i \cdot \ell_{k-1}$

$$\overline{w}^{k-1}(j) = 1/4 \cdot 1/(a_{k-1}+1) \cdot (\overline{w}^{k-1}(i \cdot \ell_{k-1} - a_{k-1}) + X^{k-1}(i \cdot \ell_{k-1} - a_{k-1}))$$
$$((\ell_{k-1} + (i-1)\ell_{k-1} + 1 - (i \cdot \ell_{k-1} - 1)) + 2(\ell_{k-1} + (i-1)\ell_{k-1} + 1 - i \cdot \ell_{k-1}))$$
$$= ((\overline{w}^{k-1}(i \cdot \ell_{k-1} - a_{k-1}) + X^{k-1}(i \cdot \ell_{k-1} - a_{k-1})))/(a_{k-1}+1)$$

The procedure $BWG(1+i \cdot \ell_{k-1}, k-1)$ generates $\overline{w}^{k-1}(j)$'s

for $j \in I((i+1)\ell_{k-1}) - I(i \cdot \ell_{k-1})$ and $i \in \{2,4,\ldots,2^{s-k+1}\}$

For $j \in I(i \cdot \ell_{k-1} + a_{k-1}) - I(i \cdot \ell_{k-1} + 1)$

$$\overline{w}^{k-1}(j) = 1/4(\overline{w}^{k-1}(j-1) + \overline{w}^{k-1}(j+1) + 2\overline{w}^{k-1}(j))$$
$$= 1/4(\overline{w}^{k}(j-1-(d_{k-1}+1)a_{k-1} - \overline{w}^{k-1}(j-1-a_{k-1})$$
$$+ \overline{w}^{k}(j+1-(d_{k-1}+1)a_{k-1}) - \overline{w}^{k-1}(j+1-a_{k-1})$$
$$+ 2\overline{w}^{k}(j-(d_{k-1}+1)a_{k-1}) - 2\overline{w}^{k-1}(j-a_{k-1}))$$
$$= 1/4(4\overline{w}^{k}(j-(d_{k-1}+1)a_{k-1}) - 4\overline{w}^{k-1}(j-a_{k-1}))$$

For $j = i \cdot \ell_{k-1} + 1$

$$\overline{w}^{k-1}(j) = 1/4(\overline{w}^{k-1}(j+1) + 2\overline{w}^{k-1}(j))$$
$$= 1/4(\overline{w}^{k}(j+1-(d_{k-1}+1)a_{k-1}) - \overline{w}^{k-1}(j+1-a_{k-1})$$
$$+ 2\overline{w}^{k}(j-(d_{k-1}+1)a_{k-1}) - 2\overline{w}^{k-1}(j-a_{k-1}))$$
$$= 1/4(4(\overline{w}^{k}(j-(d_{k-1}+1)a_{k-1}) - \overline{w}^{k}(j-a_{k-1})))$$

For $j = i \cdot \ell_{k-1} + a_{k-1} + 1$

$$\overline{w}^{k-1}(j) = 1/4(\overline{w}^{k}(j-1-(d_{k-1}+1)a_{k-1}) - \overline{w}^{k-1}(j-1-a_{k-1})$$
$$+ 2\overline{w}^{k}(j+1-(d_{k-1}+1)a_{k-1}) + w^{k-1}(j-a_{k-1}-1)$$
$$2\overline{w}^{k}(j-(d_{k-1}+1)a_{k-1}))$$
$$= \overline{w}^{k}(j-(d_{k-1}+1)a_{k-1})$$

For $j \in I((i+1)\ell_{k-1}-1) - I(i\cdot\ell_{k-1}+a_{k-1}+1)$

$$\begin{aligned}
\overline{w}^{k-1}(j) &= 1/4(\overline{w}^{k-1}(j-1) + \overline{w}^{k-1}(j+1) + 2\overline{w}^{k-1}(j)) \\
&= 1/4(\overline{w}^{k}(j-1-(d_{k-1}+1)a_{k-1}) + \overline{w}^{k}(j+1-(d_{k-1}+1)a_{k-1}) \\
&\quad + 2\overline{w}^{k}(j-(d_{k-1}+1)a_{k-1})) \\
&= \overline{w}^{k}(j-(d_{k-1}+1)a_{k-1})
\end{aligned}$$

For $j = (i+1)\ell_{k-1}$

$$\begin{aligned}
\overline{w}^{k-1}(j) &= 1/4(\overline{w}^{k-1}(j-1) + 2\overline{w}^{k-1}(j)) \\
&= 1/4(\overline{w}^{k}(i-1-(d_{k-1}+1)a_{k-1}) + 2\overline{w}^{k}(j-(d_{k-1}+1)a_{k-1})) \\
&= 1/4(\overline{w}^{k}((d_{k-1}+1)\ell_{k}-1) + 2\overline{w}^{k}((d_{k-1}+1)\ell_{k}) \\
&= \overline{w}^{k}((d_{k-1}+1)\ell_{k}) \\
&= \overline{w}^{k}((i+1)\ell_{k-1}-(d_{k-1}+1)\ell_{k})
\end{aligned}$$

In the proof, we claimed that $(i+1)\ell_{k-1} - (d_{k-1}+1)a_{k-1} = (d_{k-1}+1)\ell_{k}$. It is true because $d_{k-1}+1 = (i+1)/2$ and $\ell_{k-1} = (\ell_{k}+a_{k-1})/2$.

APPENDIX B

The closed form expressions for the cummulative weights W_i for some duplicating patterns are as follows.

1. $(0,0,\ldots,2^s-t)$

   a. $t=1$

   $W_i = (2^s+1)2^{s-i-1}$, $i = 1,2,\ldots,s-1$

   $W_s = 1$

   b. $t = 2^{s-1}$

   $W_i = 2^{s-i-1}(2^{s-1}+1)+2^{s-i-2}(2^{s-1}+1)$, $i = 1,2,\ldots,s-2$

   $W_{s-1} = 2^s+1$

   $W_s = 1$

   c. $t = 2^s-1$

   $W_i = 2^{s-i+1}$

   $W_s = 1$

2. $(0,1,1,\ldots,2^{s-1}+1)$

   $W_1 = 1/12\,(2n^2+4n-8+(-1)^s(-2))$

   $W_i = 1/(3\cdot 2^{i+1})(2n^2+4n-6+2^{2i}+2^{i+1}(-1)^{(s-i)})$, $i = 2,3,\ldots,s-1$

   $W_s = 1$

3. $(0,1,2,4,4,\ldots,7)$

   $W_1 = 256/3\cdot 2^{s-5}$

   $W_2 = 124/5+128/5(2^{s-5}+1)$

   $W_3 = 29/3+32/3(2^{s-5}+1)$

   $W_i = 16/5\cdot 2^{s-i-1}$, $i = 4,5,\ldots,s-1$

   $W_s = 1$

4. $(0,1,2,4,4,\ldots,4,0,15)$

   $W_1 = 2^{s+4}/3-32/3$

$W_2 = 2^{s+3}/5-8$

$W_3 = 2^{s+1}/3-22/3$

$W_4 = 2^s/5-11/5$

$W_i = 2^{s-i+4}/5$, $i = 5,6,\ldots,s-2$

$W_{s-1} = 32$

$W_s = 1$

5. (0,1,2,4,4,...,4,1,13)

$W_1 = 224/3\cdot 2^{s-4}$

$W_2 = 112/5\cdot 2^{s-4}-22/5$

$W_3 = 28/3\cdot 2^{s-4}-15/3$

$W_4 = 28/5\cdot 2^{s-5}-7/5$

$W_i = 14/5\cdot 2^{s-i}$, $i = 5,6,\ldots,s-2$

$W_{s-1} = 14$

$W_s = 1$

6. (0,1,2,4,4,...,4,2,11)

$W_1 = 64\cdot 2^{s-4}-1/3$

$W_2 = 96/5\cdot 2^{s-4}-7/5$

$W_3 = 8\cdot 2^{s-4}-11/3$

$W_4 = 24/5\cdot 2^{s-5}$

$W_i = 12/5\cdot 2^{s-i}$, $i = 5,6,\ldots,s-2$

$W_{s-1} = 8$

$W_s = 1$

7. (0,1,2,4,4,...,4,3,9)

$W_1 = 160/3\cdot 2^{s-4}-5/3$

$W_2 = 16\cdot 2^{s-4}+1$

$W_3 = 20/3\cdot 2^{s-4}-7/3$

$W_4 = 4 \cdot 2^{s-5}$

$W_i = 2 \cdot 2^{s-i}$, $i = 5,6,\ldots,s-2$

$W_{s-1} = 5$

$W_s = 1$

8. $(0,1,2,4,4,\ldots,4,5,5)$

$W_1 = 32 \cdot 2^{s-4}$

$W_2 = 48/5 \cdot 2^{s-4} - 6/5$

$W_3 = 4 \cdot 2^{s-4}$

$W_4 = 12/5 \cdot 2^{s-5}$

$W_i = 6/5 \cdot 2^{s-i}$, $i = 5,6,\ldots,s-2$

$W_{s-1} = 2$

$W_s = 1$

9. $(0,1,2,4,4,\ldots,4,6,3)$

$W_1 = 64/3 \cdot 2^{s-4} - 5/7$

$W_2 = 32/5 \cdot 2^{s-4} - 8/35$

$W_3 = 8/3 \cdot 2^{s-4}$

$W_4 = 8/5 \cdot 2^{s-5}$

$W_i = 4/5 \cdot 2^{s-i}$, $i = 5,6,\ldots,s-2$

$W_{s-1} = 8/7$

$W_s = 1$

10. $(0,1,2,4,4,\ldots,4,7,1)$

$W_1 = 32/3 \cdot 2^{s-4}$

$W_2 = 16/5 \cdot 2^{s-4} - 2/5$

$W_3 = 4/3 \cdot 2^{s-4} - 1/2$

$W_4 = 4/5 \cdot 2^{s-5}$

$W_i = 2/5 \cdot 2^{s-i}$, $i = 5,6,\ldots,s-2$

$W_{s-1} = 1/2$

$W_s = 1.$