Pattern Recognition, Vol. 25, No. 5, pp. 497–505, 1992

ANALYSIS OF TEMPLATE MATCHING THINNING ALGORITHMS

SRIMANTA PAL† and PRITIMOY BHATTACHARYYA‡

† Computer Aided Management Centre, Indian Institute of Management Calcutta, Joka, D.H. Road, Alipore Post Office, Post Box No. 16757, Calcutta 700 027, India

‡ Computer Science and Engineering Department, Indian Institute of Technology, Kharagpur, India

(*Received* 6 *July* 1990; *in revised form* 10 *June* 1991; *received for publication* 29 *July* 1991)

Abstract—Thinning, i.e. skeletonization and connectivity preservation, is an important operation being performed during low-level segmentation. Various template matching thinning algorithms have been proposed so far. No attempt has yet been made to conduct the average case analysis of these algorithms in order to measure their performance. In this paper a probabilistic model of average case analysis of template matching thinning algorithms is proposed. Using the proposed model of analysis a bound on the number of iterations required is computed and also the requirement of average time to complete the process of thinning of a uniformly distributed binary image in sequential as well as a parallel environment. Also a mathematical function is proposed to compute the number of cycles involved in the thinning process originated by a given algorithm.

Thinning   Binary image   Template

1. INTRODUCTION

In computer vision, thinning, i.e. skeletonization and connectivity preservation, is an important task to be performed during low-level segmentation. We consider here template matching thinning algorithms in which criteria for skeletonization and connectivity preservation are realized in the form of templates. Hence the method of skeletonization and connectivity preservation is reduced to the template matching method. A set of templates is applied on an edge image to check whether the desired matching criteria are fulfilled.

A series of template matching thinning algorithms have been proposed in the last two decades. Some recent thinning algorithms, parallel and sequential with one-pass and multi-pass have been discussed in references (1–8). Zhang and Suen[1] proposed a parallel thinning algorithm with two sub-iterations of which one aimed at deleting the south-east boundary points and north-west corner points while the other aimed at deleting the north-west boundary points and the south-east corner points. In their algorithm, connectivity of pixels is preserved and each pattern is thinned down to a skeleton of unity thickness. During formulation of the algorithm they considered the number of 1s of its 8-neighbour (3×3 window) in the binary pattern would be between 2 and 6. Lu and Wang[2] observed that the number of 1s of its 8-neighbour in the binary pattern was between 3 and 6, and also highlighted few disadvantages, namely, (i) preservation of noise propagating pixels, (ii) distortion of shape and (iii) total disappearance of some digital patterns. Up to now this algorithm is better with respect to the other currently available 2-pass thinning algorithms. Holt *et al.*[3] improved the above algorithm[1] with respect to the time complexity and the number of passes. In this algorithm,[3] a vertical stroke of width 2 is guarded by keeping one of its edges and also maintaining the west-ward bias of the original algorithm. An element on a west edge is preserved if it is not on a corner and its east neighbour is on an edge, i.e. edge(East) and value(North) and value(South) must be true. Similarly, the north edge of a horizontal stroke of width 2 is preserved if edge(South) and value(East) and value(West) are true. Also the removal of each element of a 2×2 square can be prevented by checking the east, south and south-east neighbours, i.e. edge(East) and edge(South-East) and edge(South) are true and also they included the condition for at least one absent neighbour. This algorithm does not always preserve the skeleton of the binary pattern according to the shape but it ensures the connectedness. In this parallel, one-pass algorithm they actually considered a 4×4 window. Another one-pass thinning algorithm by Chain *et al.*[4] who considered a different set of templates, mainly (i) eight thinning templates of 3×3 windows, (ii) two restoring templates of size 4×4 window and (iii) eight trimming templates of 3×3 window. The main disadvantage of this algorithm is distortion of the skeleton of the binary output pattern.

All the thinning algorithms, mentioned previously, are basically template matching procedures based on which deletable edge points can be removed.

Though worst-case analysis of thinning algorithms has been conducted[3,8] but no attempt has ever been

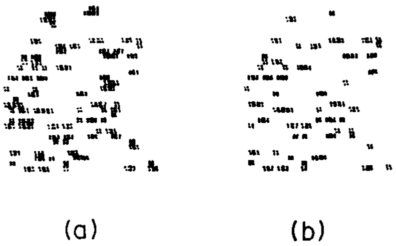

Fig. 1. Uniformly distributed random binary image: (a) original; (b) thinned.

made to conduct an average case analysis of the same. In this paper, we have attempted to formulate the average case analysis of the problem. This formulation of average case analysis is based on an observation that the pattern-matching problem is analogous to the urn model problem with multiple colour balls but without replacement. As the template matching thinning operation is a local operation being performed in the absence of a priori knowledge regarding the shape of the boundary to be thinned, it is not guided by any specific distribution of 0–1s. Hence, for the sake of simplicity it may be assumed that during average-case analysis of the thinning algorithm 0–1s are uniformly distributed.

However, since a uniformly distributed random image may not always contain 4- or 8-connected components (Fig. 1) during skeletonization, it would be difficult to obtain skeletons in that image. In order to avoid this situation we have considered here only those uniformly distributed random binary images which contain 4-connected simple objects (components). The random images with 4-connected simple objects have been generated in such a manner that for each non-zero point P in the image there exists at least one non-zero point Q such that P and Q are connected in a non-empty subset S of the image. This simple criterion of connectivity between two points P and Q has been considered by assuming the existence of a 4-path $P = P_0, P_1, \ldots, P_m = Q$ from P to Q, where P_i is 4-adjacent to P_{i-1}, $1 \le i \le n$. Figure 2 illustrates a few random images with 4-connected simple objects.

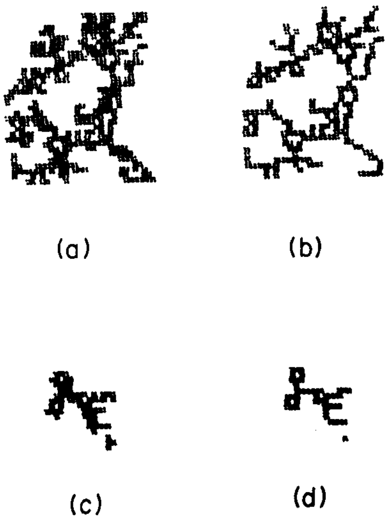

Fig. 2. Uniformly distributed random binary images containing 4-connected objects: (a), (c) original; (b), (d) thinned.

2. AVERAGE CASE ANALYSIS

2.1. *A probabilistic formulation (an urn model problem)*

The probabilistic analysis of a thinning algorithm can be viewed as an urn model problem.[9,10] The formulation is as follows.

We may use a colour to denote each decision template for deletion of the central element. Each element of the binary image can be considered to be a ball and its colour is computed from the combinations of patterns of its neighbours of size of the given templates. Let us consider the following notations:

$u \times v$ = size of the window of the template (Fig. 3) used for thinning
$w = u \times v$ = number of elements in the template
2^w = number of different patterns of the window (all possible templates) produced or
2^w = number of different colours which exist in this process.

Now we can divide these 2^w colours into two basic classes because the given pattern is a binary pattern. The colour of one class as the central element of the templates produced from the given window is zero, defined as 0-colour class. Similarly the colour of the other class as the central element of the templates produced from the given window is one, defined as 1-colour class (Fig. 4). The central element of the

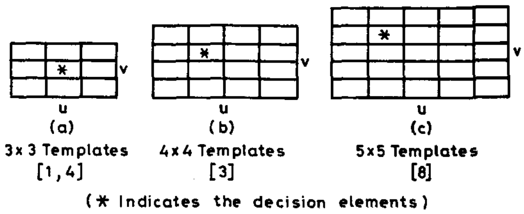

Fig. 3. Templates of size $w = u \times v$.

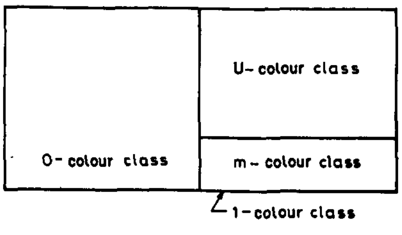

Fig. 4. Colour classes generated from the templates of an algorithm.

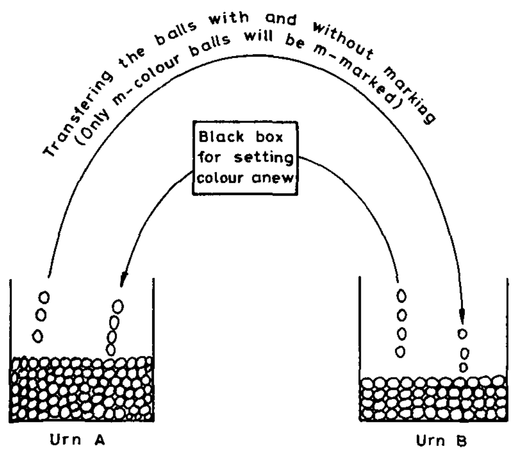

Fig. 5. Thinning process viewed as an urn model problem.

thinning templates is considered as 1 in all the discussed thinning algorithms and also the object of a thinning algorithm is to replace 1 by 0 as discussed earlier. Each element of a binary pattern is considered as a central element of a template. If the central element matches with a given set of thinning templates then this element is converted to 0 from 1. This converted value is to be used in the subsequent iteration, and this process is continued (see Fig. 5).

Let us consider, for an iteration, each element of a binary image is a ball which has a colour as computed above:

$2^{(w-1)}$ = number of colours in 0- and 1-colour class produced by the templates of a given window size
n_0 = number of colours in 0-colour class produced by the templates of a given window size = $2^{(w-1)}$.

Again the 1-colour class can be divided into two sub-classes of which one sub-class is the match-colour (m-colour) class and the other is the unmatch-colour (u-colour) class (see Fig. 4).

Let us assume the following:

n_m = number of colours in the m-colour class, i.e. number of templates used for thinning
n_u = number of colours in the u-colour class, i.e. number of generated templates whose centre is 1 but not used for thinning = $2^{(w-1)} - n_m$.

As the colours in each class or sub-class are equally likely and independent, we can consider the following probabilities:

p_0 = probability of a colour belongs to 0-colour class = $2^{(w-1)}/2^w = 1/2$
p_1 = probability of a colour belongs to 1-colour class = $2^{(w-1)}/2^w = 1/2$
p_m = probability of a colour belongs to m-colour class = $n_m/2^w$
p_u = probability of a colour belongs to u-colour class = $1/2 - p_m$

p'_m = probability of a colour belongs to m-colour class under the condition that it also belongs to 1-colour class = $n_m/2^{(w-1)}$
p'_u = probability of a colour belongs to u-colour class under the condition that it also belongs to 1-colour class = $1 - p'_m$.

We can write: (1) $p_0 + p_1 = 1$, (2) $p_0 + p_m + p_u = 1$, and (3) $p'_m + p'_u = 1$.

Consider a binary image of size $n \times m$ which is to be thinned by using a given one-pass thinning algorithm. The number of elements in the image is $N = n \times m$.

Theorem 1 (*urn model*). There are N balls kept in an Urn A. Each ball has one of the three colours, namely, 0-colour, m-colour or u-colour. The process starts as follows. A ball is drawn randomly from Urn A and if it is m-colour then the ball is marked and placed into another Urn B. If the ball is not of m-colour then it is simply placed into Urn B without marking. This process continues until the cycle is completed. A cycle is completed when all the balls of Urn A have been transferred to Urn B. At the end of a cycle the balls of Urn B are passed through a black box (Fig. 5). Each ball is coloured anew there and then transferred into Urn A. The above process is repeated in the subsequent cycles. The whole process is terminated when no m-colour ball is found

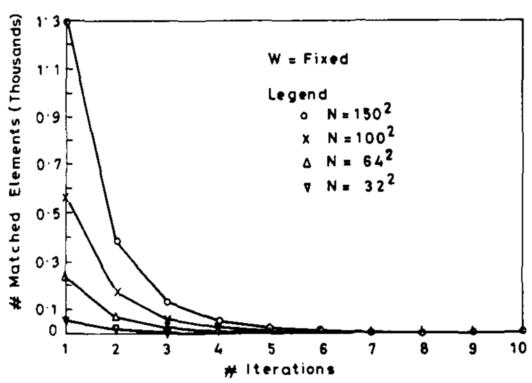

Fig. 6. Distribution of matched elements.

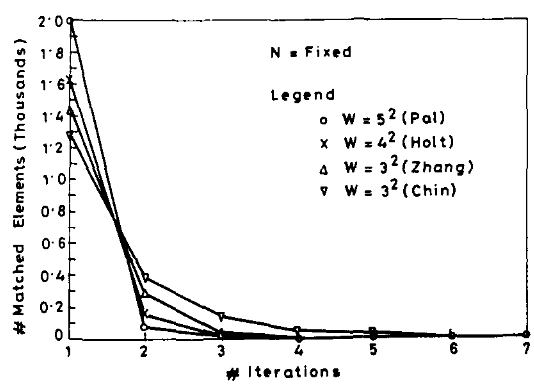

Fig. 7. Distribution of matched elements.

in a complete cycle. If the number of cycles in the above process is k, under the assumptions:

Assumption I

(a) all the m-colour balls converted to 0-colour balls and

(b) some of the remaining 1-colour balls converted to m-colour and u-colour balls depend upon the probability distribution, the bounds of k are

$$1 \le k \le \log(2/(Np'_m))/\log(1-p'_m).$$

Assumption II

(a) all the m-colour balls converted to 0-colour balls and

(b) some of the u-colour balls converted to m-colour balls depend upon the probability distribution, the bounds of k are

$$1 \le k \le \log(2/N)/\log(2p_m) - 1.$$

2.2. *Proof of Theorem* 1

Case I: proof under assumption I. Since the image is uniformly distributed over 0 and 1 we can consider the following. At the beginning of the first cycle Urn A contains:

total number of balls $= N$
total number of 0-colour balls $= N_0^{(0)} = Np_0 = N/2$
total number of 1-colour balls $= N_1^{(0)} = Np_1 = N/2$
total number of m-colour balls $= N_m^{(0)} = N_1^{(0)}p'_m$
total number of u-colour balls $= N_u^{(0)} = N_1^{(0)}p'_u$.

Therefore, $N = N_0^{(0)} + N_m^{(0)} + N_u^{(0)}$.

Lemma 1 (*terminating criterion*). At the end of thinning after the kth iteration the number of m-colour balls will be less than one, i.e. $N_m^{(k)} < 1$ and $N_0^{(k)} + N_0^{(k)} > N - 1$ where

$N_m^{(k)} =$ number of m-colour balls after kth iteration
$N_0^{(k)} =$ number of 0-colour balls after kth iteration
$N_0^{(k)} =$ number of u-colour balls after kth iteration.

Proof. Since $N_m^{(k)}$ is the number of m-colour balls after the kth iteration and k iterations are required for thinning an image, this implies that $N_m^{(k)} = 0$, i.e. $N_m^{(k)} < 1$ and $N_0^{(k)} + N_0^{(k)} > N - 1$. At the end of the kth iteration the number of m-colour balls will be less than or equal to one. Therefore, $N_m^{(k)} \le 1$ and $N_0^{(k)} + N_u^{(k)} \ge N - 1$.

Since the sample is uniformly distributed over the binary value (0, 1) the thinning algorithm converts $N_m^{(0)}$ m-colour balls into 0-colour balls. So at the end of the first cycle the number of 0-colour balls will be $N_0^{(1)} = N_0^{(0)} + N_m^{(0)}$ and the number of 1-colour balls will be $N_1^{(1)} = N_u^{(0)}$. The effect of conversion from m-colour to 0-colour is that some of the $N_u^{(0)}$ unmatched balls will be converted to m-colour as $N_m^{(1)} = N_1^{(1)}p'_m$ and $N_u^{(1)} = N_1^{(1)}p'_u$. But the total number of balls will remain unchanged at the end of the first iteration, i.e. when $N = N_0^{(1)} + N_m^{(1)} + N_u^{(1)}$.

Lemma 2. $N = N_0^{(k)} + N_m^{(k)} + N_u^{(k)}$ where

$$N_0^{(k)} = N_0^{(k-1)} + N_m^{(k-1)} = N - N(p'_u)^k/2$$
$$N_1^{(k)} = N_u^{(k-1)}$$
$$N_m^{(k)} = N_1^{(k)}p'_m = N(p'_u)^k p'_m/2$$

and

$$N_u^{(k)} = N_1^{(k)}p'_u = N(p'_u)^{k+1}/2.$$

Proof. Any m-colour element (ball) affects its non-zero 8-neighbours because it is converted into a 0-colour element. Furthermore, each m-colour ball will contribute 1 to neighbouring u-colour balls resulting in the same transformation of u-colour into m-colour balls. Therefore, at the end of the first iteration

$$\begin{aligned} N_0^{(1)} &= N_0^{(0)} + N_m^{(0)} = N/2 + Np'_m/2 \\ &= N(1+p'_m)/2 = N - N(p'_u)^1/2 \\ N_1^{(1)} &= N_0^{(0)} = Np'_u/2 \\ N_m^{(1)} &= N_1^{(1)}p'_m = N(p'_u)^1 p'_m/2 \\ N_u^{(1)} &= N_1^{(1)}p'_u = N(p'_u)^2/2. \end{aligned}$$

The total number of balls does not change, so

$$N = N_0^{(1)} + N_m^{(1)} + N_u^{(1)}.$$

Again, at the end of the second iteration

$$\begin{aligned} N_0^{(2)} &= N_0^{(1)} + N_m^{(1)} = N(1 + p'_m + (p'_u)^1 p'_m)/2 \\ &= N - N(p'_u)^2/2 \\ N_1^{(2)} &= N_u^{(1)} = N(p'_u)^2/2 \\ N_m^{(2)} &= N_1^{(2)}p'_m = N(p'_u)^2 p'_m/2 \\ N_u^{(2)} &= N_1^{(2)}p'_u = N(p'_u)^3/2. \end{aligned}$$

By the conservation rule (the total number of balls does not change)

$$N = N_0^{(2)} + N_m^{(2)} + N_u^{(2)}.$$

So by mathematical induction we can get at the end of the kth iteration

$$\begin{aligned} N_0^{(k)} &= N_0^{(k-1)} + N_m^{(k-1)} = N(1 + p'_m \\ &\quad + [(p'_u)^1 + (p'_u)^2 + \cdots + (p'_u)^{k-1}]p'_m)/2 \\ &= N - N(p'_u)^k/2 \\ N_1^{(k)} &= N_u^{(k-1)} \\ N_m^{(k)} &= N_1^{(k)}p'_m = N(p'_u)^k p'_m/2 \end{aligned}$$

and

$$N_u^{(k)} = N_1^{(k)}p'_u = N(p'_u)^{k+1}/2.$$

Again, by the conservation rule

$$N = N_0^{(k)} + N_m^{(k)} + N_u^{(k)}.$$

By lemma 1 and lemma 2, we can write

$$N(p'_u)^k p'_m/2 < 1 \text{ or } N[2-(p'_u)^k]/2 + N(p'_u)^{k+1}/2$$
$$> N - 1$$
$$\Rightarrow (p'_u)^k < 2/(Np'_m) \Rightarrow k < \log(2/(Np'_m))/\log(p'_u).$$

Therefore, the average number of cycles (iterations) is

$$k = \log(2/(Np'_m))/\log(1 - p'_m).$$

Hence, the bounds of k are

$$1 \le k \le \log(2/(Np'_m))/\log(1 - p'_m).$$

Case II: proof under assumption II. Since the image is uniformly distributed over 0 and 1 we can consider the following. At the beginning of the first cycle Urn A contains:

total number of balls $= N$
total number of balls $= N_0^{(0)} = Np_0 = N/2$
total number of m-colour balls $= N_m^{(0)} = Np_m = Nn_m/2^w = Np'_m/2$
total number of u-colour balls $= N_u^{(0)} \approx Np_u = Np'_u/2$.

Therefore, $N = N_0^{(0)} + N_m^{(0)} + N_u^{(0)}$.

Since the sample is uniformly distributed over the binary value (0, 1) the thinning algorithm converts $N_m^{(0)}$ m-colour balls into 0-colour balls. So at the end of the first cycle the number of 0-colour balls will be $N_0^{(1)} = N_0^{(0)} + N_m^{(0)}$. The effects of conversion from m-colour to 0-colour is that some of the $N_u^{(0)}$ unmatched balls will be converted to m-colour. But the total number of balls will remain unchanged at the end of the first iteration, i.e. when $N = N_0^{(1)} + N_m^{(1)} + N_u^{(1)}$.

Lemma 3. $N = N_0^{(k)} + N_m^{(k)} + N_u^{(k)}$ where

$$N_0^{(k)} = N_0^{(k-1)} + N_m^{(k-1)}$$
$$N_m^{(k)} = N_m^{(k-1)}p'_m = N_m^{(0)}(p'_m)^k$$
$$N_u^{(k)} = (N_u^{(k-1)} - N_m^{(k-1)}) + N_m^{(k-1)}p'_u$$
$$= N_u^{(k-1)} - N_m^{(k-1)}p'_m.$$

Proof. Any m-colour element (ball) affects its non-zero 8-neighbours because it is converted into a 0-colour element. Furthermore, each m-colour ball will contribute 1 to neighbouring u-colour balls resulting in the same transformation of u-colour into m-colour balls. Therefore, at the end of the first iteration

$$N_0^{(1)} = N_0^{(0)} + N_m^{(0)}$$
$$N_m^{(1)} = N_m^{(0)}p'_m = N_m^{(0)}(p'_m)^1$$
$$N_u^{(1)} = (N_u^{(0)} - N_m^{(0)}) + N_m^{(0)}p'_u = N_u^{(0)} - N_m^{(0)}p'_m.$$

The total number of balls does not change, so

$$N = N_0^{(1)} + N_m^{(1)} + N_u^{(1)}.$$

Again, at the end of the second iteration

$$N_0^{(2)} = N_0^{(1)} + N_m^{(1)}$$
$$N_m^{(2)} = N_m^{(1)}p'_m = N_m^{(0)}(p'_m)^2$$
$$N_u^{(2)} = (N_u^{(1)} - N_m^{(1)}) + N_m^{(1)}p'_u = N_u^{(1)} - N_m^{(1)}p'_m.$$

By the conservation rule (the total number of balls does not change)

$$N = N_0^{(2)} + N_m^{(2)} + N_u^{(2)}.$$

So by mathematical induction we can get at the end of the kth iteration

$$N_0^{(k)} = N_0^{(k-1)} + N_m^{(k-1)}$$
$$N_m^{(k)} = N_m^{(k-1)}p'_m = N_m^{(0)}(p'_m)^k$$
$$N_u^{(k)} = (N_u^{(k-1)} - N_m^{(k-1)}) + N_m^{(k-1)}p'_u$$
$$= N_u^{(k-1)} - N_m^{(k-1)}p'_m.$$

Again, by the conservation rule $N = N_0^{(k)} + N_m^{(k)} + N_u^{(k)}$.

Lemma 4. $N_0^{(k)} + N_u^{(k)} = N\{2 - (p'_m)^{k+1}\}/2$ and $N_m^{(k)} = N(p'_m)^{k+1}/2$.

Proof. From the above results we get

$$N_0^{(0)} = N/2$$
$$N_m^{(0)} = Np'_m/2$$
$$N_u^{(0)} = Np'_u/2$$

for $k = 1$

$$N_0^{(1)} = N_0^{(0)} + N_m^{(0)} = N(1 + p'_m)/2$$
$$N_m^{(1)} = N_m^{(0)}p'_m = Np'_m p'_m/2 = N(p'_m)^2/2$$
$$N_u^{(1)} = N_u^{(0)} - N_m^{(0)}p'_m = Np'_u/2 - (Np'_m/2)p'_m$$
$$= N(1 - p'_m)/2 - (Np'_m/2)p'_m$$
$$= N[1 - p'_m - (p'_m)^2]/2$$

for $k = 2$

$$N_0^{(2)} = N_0^{(1)} + N_m^{(1)} = N(1 + p'_m)/2 + N(p'_m)^2/2$$
$$= N(1 + p'_m + (p'_m)^2)/2$$
$$N_m^{(2)} = N_m^{(1)}p'_m = Np'_m(p'_m)^2/2 = N(p'_m)^3/2$$
$$N_u^{(2)} = N_u^{(1)} - N_m^{(1)}p'_m$$
$$= N(1 - p'_m - (p'_m)^2)/2 - (N(p'_m)^2)/2p'_m$$
$$= N(1 - p'_m - (p'_m)^2 - (p'_m)^3)/2.$$

Therefore, by mathematical induction, we can write

$$N_0^{(k)} = N(1 + p'_m + (p'_m)^2 + \cdots + (p'_m)^k)/2$$
$$N_m^{(k)} = N(p'_m)^{k+1}/2$$
$$N_u^{(k)} = N(1 - p'_m - (p'_m)^2 - (p'_m)^3 - \cdots$$
$$- (p'_m)^{k+1})/2$$

and

$$N_0^{(k)} + N_u^{(k)} = N(2 - (p'_m)^{k+1})/2.$$

By lemma 1 and lemma 4, we can write

$$N(p'_m)^{k+1}/2 < 1 \quad \text{and} \quad N(2 - (p'_m)^{k+1})/2 > N - 1$$
$$\Rightarrow (p'_m)^{k+1} < 2/N$$
$$\Rightarrow k + 1 < [\log(2/N)]/[\log p'_m].$$

Therefore, the average number of cycles (iterations) is

$$k = [\log(2/N)]/[\log p'_m] - 1.$$

Hence, the bounds of k are

$$1 \le k \le \log(2/N)/\log(2p_m) - 1.$$

2.3. *Average time complexity*

As shown in the previous sections that the proposed urn model formulation can be used to compute the average number of iterations required in the case of a given thinning algorithm and in each iteration the average number of elements to be converted from non-zero (1) to zero elements. Now these results will be used to compute the average time requirement of the given algorithm. Let us assume the following:

b = unit of time taken for a boolean operation, assignment or if-then-else
s = unit of time taken to match or mismatch the thinning templates of the thinning algorithm.

2.3.1. *Sequential.* We consider here that the parallel template matching thinning algorithms will be executed sequentially.

Case I. Using assumption I of Theorem 1 the time required for the first iteration is computed as follows:

$$T_1 = bN/2 + sN/2 = bN_0^{(0)} + sN_1^{(0)}.$$

Similarly, for the next kth iterations we can write

$$T_k = bN_0^{(k-1)} + sN_1^{(k-1)} = b(N - N(p'_u)^{k-1}/2) + sN(p'_u)^{k-1}/2 = Nb + (s-b)N(p'_u)^{k-1}/2.$$

Therefore, the total time after the kth iteration is

$$T = T_1 + T_2 + T_3 + \cdots + T_k$$
$$= kNb + N(s-b)\,\Sigma_{1\le i\le k}\,(p'_u)^{i-1}/2$$
$$= kNb + N(s-b)[1-(p'_u)^k]/(2p'_m) = O(Nk).$$

Case II. Using assumption II of Theorem 1 the time required for the first iteration is computed as follows:

$$T_1 = bN_0^{(0)} + s(N_m^{(0)} + N_u^{(0)}) = bN/2 + sN/2.$$

Similarly, for the next kth iterations we can write $T_1 = bN_0^{(k-1)} + s(N_m^{(k-1)} + N_u^{(k-1)})$. Therefore, the total time after the kth iteration is

$$T = T_1 + T_2 + T_3 + \cdots + T_k$$
$$= b\Sigma_{1\le i\le k}N_0^{(i-1)} + s\Sigma_{1\le i\le k}(N_m^{(i-1)} + N_u^{(i-1)})$$
$$N_0^{(i)} = N(1 + p'_m + (p'_m)^2 + \cdots + (p'_m)^i)/2$$
$$= N(1-(p'_m)^{i+1})/[2(1-p'_m)].$$

So

$$\Sigma_{1\le i<k}N_0^{(i)} = \Sigma_{1\le i<k}N(1-(p'_m)^{i+1})/[2(1-p'_m)]$$
$$= N[k - p'_m\{(1-(p'_m)^k)/(1-p'_m)\}]/[2(1-p'_m)]$$
$$N_m^{(i)} + N_u^{(i)} = N[2 - \{1-(p'_m)^{i+1}\}/(1-p'_m)]/2$$

and

$$\Sigma_{1\le i\le k}(N_m^{(i)} + N_u^{(i)}) = N[2k - \{k - p'_m(1 - (p'_m)^k)/(1-p'_m)\}/(1-p'_m)]/2.$$

Therefore, total time

$$T = bN[k(1-p'_m) - p'_m(1-(p'_m)^k)]/[2(1-p'_m)^2] + sN[2k(1-p'_m)^2 - k(1-p'_m) + p'_m(1-(p'_m)^k)]/[2(1-p'_m)^2]$$
$$= N[\{k(1-p'_m) - p'_m(1-(p'_m)^k)\}b + \{k(1-p'_m)(1-2p'_m) + p'_m(1-(p'_m)^k)\}s]/[2(1-p'_m)^2]$$
$$= N[k(1-p'_m)(b-(1-2p'_m)s) + p'_m(1-(p'_m)^k)(s-b)]/[2(1-p'_m)^2]$$
$$= O(Nk).$$

2.3.2. *Parallel.* In the case of parallel execution, let us consider the following:

$N = r \times c$ = number of binary elements in the given image
r = number of rows of the binary image
c = number of columns of the binary image
p = number of processors used for thinning.

Case I. Using assumption I of Theorem 1 the average time taken for c elements after the kth iteration is computed as follows: $T_i = c[kb + (s-b)\{1-(p'_u)^k\}/(2p'_m)]$, for $i = 1, 2, \ldots, r$. Therefore, the total time taken in the parallel computation is $T = \lfloor 1 + (r-1)/p \rfloor [\mathrm{Max}_{1\le i\le r} T_i]$. Therefore, the total average time taken is $T = \lfloor 1 + (r-1)/p \rfloor * c[kb + (s-b)\{1-(p'_u)^k\}/(2p'_m)] = O(Nk/p)$.

Case II. Using assumption II of Theorem 4.1 the average time taken for c elements after the kth iteration is computed as follows:

$$T_i = c[k(1-p'_m)(b + (1-2p'_m)s) + p'_m(1-(p'_m)^k)(s-b)]/[2(1-p'_m)^2], \quad \text{for } i = 1, 2, \ldots, r.$$

Therefore, the total time taken after the kth iteration by the parallel computation is

$$T = \lfloor 1 + (r-1)/p \rfloor [\mathrm{Max}_{1\le i\le r} T_i].$$

Therefore, the total average time taken

$$T = \lfloor 1 + (r-1)/p \rfloor c[k(1-p'_m)(b + (1-2p'_m)s) + p'_m(1-(p'_m)^k)(s-b)]/[2(1-p'_m)^2]$$
$$= O(Nk/p).$$

Table 1. Parameters of $f(t)$ for different thinning algorithms

Algorithms	$p \times 100$	Constants b	c	d	w
Zhang and Suen [1]	6.64	6.64	−0.085	−0.010	3
Chain *et al.* [4]	9.37	$9.37 \times 2^{t-1}$	0.180	−0.020	3
Holt *et al.* [3]	7.36	7.36	−0.001	−0.070	4
Pal and Bhattacharyya [8]	7.78	7.78	0.002	−0.060	5

3. APPROXIMATED BEHAVIOUR

The experimental results under the proposed average case analysis model are shown in Table 8. Observing the experimental results a negative exponential function $f(t)$ can be proposed to find the possible number of matching at the tth iteration. The following form of $f(t)$ is obtained:

$$f(t) = b \exp\left(-\sqrt{wt + d(1+t)^2 + c}\right)$$

where b is a constant in the case of the algorithms due to Zhang and Suen,[1] and Holt *et al.*[3] and the modified algorithm of Holt.[8] b is a function of t in the case of Chain *et al.*'s [4] algorithm. c and d are constants for all the algorithms. The values of the constants b, c and d are shown in Table 1.

The above mathematical function has been used to compute the percentage (%) of matched elements occurring at different iterations. These results are shown in Table 9. It is interesting to note that the computed results (Table 9) are very close to the experimental results (Table 8).

4. RESULTS AND CONCLUSION

We are presenting here the results obtained by evaluating the average performance experimentally as well as the help of the proposed average case analysis model. The objective is to compare the experimental results with the calculated one.

It has been found experimentally that the number of iterations depends on the number of elements presented in the thinning templates used in the algorithm. In our experiments we have considered four algorithms, namely, those due to Zhang and Suen,[1] Chain *et al.*,[4] Holt *et al.*[3] and the modified Holt algorithm.[8]

Table 2 illustrates the number of templates used and the probability of template matching in the case of the above four algorithms.

The average number of iterations required for images with different sizes are shown in Table 3. The average number of iterations has been calculated using our proposed formulation for average case analysis. The formulation under assumption I (Case I) has been used in this case.

The proposed formulation under assumption II (Case II) has been used to compute the average number of iterations required for thinning. These results are shown in Table 4.

As shown in Table 5 the average numbers of iterations required for thinning have been obtained experimentally for images of different sizes.

Tables 6 and 7 display the percentage (%) of

Table 2. The number of templates and probability of matching

Algorithm used	Number of elements in a template, w	Number of templates used, n_m	Probability of template matching, $p_m = n_m/2^w$
Zhang and Suen	9	34	0.06640625
Chain *et al.*	9	48	0.09375000
Holt *et al.*	16	4823	0.07359314
Pal and Bhattacharyya	25	2,608,640	0.07774353

Table 3. Average number of iterations required for images with different sizes (calculated Case I)

Image size	Zhang and Suen	Chain *et al.*	Holt *et al.*	Pal and Bhattacharyya
32 × 32	29.61	21.98	27.15	26.97
64 × 64	39.34	28.66	35.85	35.60
100 × 100	45.60	32.96	41.46	41.16
150 × 150	51.29	36.86	46.55	46.22

Table 4. Average number of iterations required for images with different sizes (calculated Case II)

Image size	Zhang and Suen	Chain *et al.*	Holt *et al.*	Pal and Bhattacharyya
32 × 32	2.09	2.73	2.26	2.35
64 × 64	2.78	3.55	2.98	3.10
100 × 100	3.22	4.09	3.44	3.58
150 × 150	3.62	4.57	3.87	4.01

Table 5. Average number of iterations required for images with different sizes (experimental)

Image size	Zhang and Suen	Chain *et al.*	Holt *et al.*	Pal and Bhattacharyya
32 × 32	2.0	4.30	2.7	2.60
64 × 64	2.1	7.20	3.0	3.30
100 × 100	2.3	7.50	3.4	3.80
150 × 150	2.6	8.40	4.2	4.10

Table 6. Average percentage of matched elements in the cycles (calculated Case I)

				Cycle			
Algorithm	1	2	3	4	5	6	7
Zhang and Suen	6.64	5.76	4.9938	4.330	3.7554	3.2567	2.8
Chain *et al.*	9.37	7.62	6.1889	5.028	4.0857	3.3196	2.7
Holt *et al.*	7.36	6.27	5.3523	4.564	3.8927	3.3198	2.8
Pal and Bhattacharyya	7.78	6.56	5.5447	4.682	3.9545	3.3396	2.8

Table 7. Average percentage of matched elements in the cycles (calculated Case II)

				Cycle			
Algorithm	1	2	3	4	5	6	7
Zhang and Suen	6.64	0.44	0.0293	0.002	0.00013	0.00000	0.00
Chain *et al.*	9.37	0.88	0.0824	0.008	0.00072	0.00007	0.00
Holt *et al.*	7.36	0.54	0.0398	0.003	0.00022	0.00002	0.00
Pal and Bhattacharyya	7.78	0.60	0.0470	0.004	0.00028	0.00002	0.00

Table 8. Average percentage of matched elements in the cycles (experimental)

				Cycle			
Algorithm	1	2	3	4	5	6	7
Zhang and Suen	6.09198	1.16533	0.17116	0.03494	0.00641	0.00083	0.00000
Chain *et al.*	5.50592	1.58898	0.53514	0.19480	0.06985	0.02488	0.00922
Holt *et al.*	6.79578	0.67540	0.07249	0.00883	0.00058	0.00011	0.00000
Pal and Bhattacharyya	7.30087	0.61790	0.07217	0.00924	0.00022	0.00000	0.00000

matched elements encountered at different iterations. The results have been computed by using both formulations, namely, Cases I and II.

The percentage of matched elements present in different iterations under different algorithms has also been obtained experimentally. These are reported in Table 8.

According to Tables 3–8 it is apparent that the experimental results shown in Tables 5 and 8 are close to the computed results shown in Tables 4 and

Table 9. Average percentage of matched elements in the cycles (behavioural)

Algorithm	Cycle 1	2	3	4	5	6	7
Zhang and Suen	6.03824	1.03672	0.17447	0.02878	0.00465	0.00074	0.0001
Chain *et al.*	5.49790	1.83210	0.58658	0.18044	0.05333	0.01514	0.0041
Holt *et al.*	6.85555	0.75206	0.07172	0.00595	0.00043	0.00003	0.0000
Pal and Bhattacharyya	7.34160	0.65540	0.05189	0.00364	0.00023	0.00001	0.0000

7, respectively. It is apparent that the experimental results (Tables 5 and 8) are deviating from the computed results (Tables 3 and 6) obtained under assumption I (Case I) whereas the computed results (Tables 4 and 7) obtained under assumption II (Case II) are close to the experimental results.

While conducting the experiment it has been observed that the distribution pattern of matching templates is a decaying process represented by a near exponential distribution (Section 3 and Table 9) with the image size and the template size being the parameters and constants depending upon the algorithms chosen as shown in Figs 6 and 7. Work is also in progress to obtain a generalized probabilistic model for average case analysis of the thinning algorithm.

Acknowledgements—The authors would like to thank the reviewers for their valuable comments and necessary suggestions towards the improvement of the paper.

REFERENCES

1. T. Y. Zhang and C. Y. Suen, A fast parallel algorithm for thinning digital patterns, *CACM* **27**(3), 236–239 (1984).
2. H. E. Lu and S. P. Wang, A comment on 'A fast parallel algorithm for thinning digital patterns', *CACM* **29**(3), 239–242 (1986).
3. C. M. Holt, A. Stewart, M. Clint and R. H. Pervolt, An improved parallel thinning algorithm, *CACM* **30**(2), 156–160 (1987).
4. R. T. Chain, H. K. Wan, D. L. Stover and R. D. Iverson, A one-pass thinning algorithm and its parallel implementation, *Comput. Graphics Image Process.* **40**(1), 30–40 (1987).
5. E. R. Davies and A. P. N. Plemmer, Thinning algorithms: a critique and a new methodology, *Pattern Recognition* **14**, 53–63 (1981).
6. S. Suzuki and K. Abe, Binary picture thinning by an iterative parallel two-sub-cycle operation, *Pattern Recognition* **20**, 297–307 (1987).
7. V. K. Govindan and A. P. Srivaprasad, A pattern adaptive thinning algorithm, *Pattern Recognition* **20**, 623–637 (1987).
8. S. Pal and P. Bhattacharyya, A shape preserving one-pass parallel thinning algorithm, Indian Institute of Management Calcutta, Working Paper Series, WPS-123(89) (1989).
9. W. Filler, *An Introduction to Probability Theory and its Applications*, Vol. 1. Wiley Eastern (1968).
10. N. L. Johnson and S. Kotz, *Urn Models and their Application: An Approach to Modern Discrete Probability Theory.* Wiley, New York (1977).
11. S. G. Akl, *The Design and Analysis of Parallel Algorithms.* Prentice-Hall, Englewood Cliffs, New Jersey (1989).

About the Author—SRIMANTA PAL received his B.Tech. degree in instrumentation and electronics from Jadavpur University in 1982, M.Tech. degree in computer science from Indian Statistical Institute Calcutta in 1984 and M.B.A. degree from Jadavpur University in 1988. Mr Pal then joined Semiconductor Complex Ltd at Chandigarh (India) as an engineer. From March 1987 to June 1991 he worked as a System Analyst at the Computer Aided Management Centre in Indian Institute of Management Calcutta. Recently Mr Pal has joined Indian Statistical Institute Calcutta and has been working there as a System Analyst. His areas of interest are computer vision, computational geometry and design, and analysis of algorithms.

About the Author—PRITIMOY BHATTACHARYYA received his M.Sc. and M.Phill. (DIIT) degrees in mathematics from Indian Institute of Technology (IIT) Kharagpur. In 1985, he did his Ph.D. in image processing from IIT Kharagpur. Dr Bhattacharyya joined the Department of Computer Science and Engineering, IIT Kharagpur in February 1978 as a Lecturer. He has been working there as an Associate Professor since July 1991. From July 1987 to June 1988, he worked as a visiting Assistant Professor at the Computer Aided Management Centre, Indian Institute of Management Calcutta. Dr Bhattacharyya is a senior member of IEEE and a member of the Editorial Board of CSI journal *Computer Science and Informatics*. His areas of interest are computer vision, OS design, DBMS and distributed systems. Dr Bhattacharyya is the co-author of the book *Database Design Principles* which is being published by Tata McGraw-Hill. He has been active in design and teaching courses in Information Technology for various Public and Private Sector undertakings over the country.