On Shrinking Binary Picture Patterns

S. Levialdi
Laboratorio di Cibernetica del C.N.R.
Napoli, Italy

A parallel processing algorithm for shrinking binary patterns to obtain single isolated elements, one for each pattern, is presented. This procedure may be used for counting patterns on a matrix, and a hardware implementation of the algorithm using large scale integrated technology is envisioned. The principal features of this method are the very small window employed (two-by-two elements), the parallel nature of the process, and the possibility of shrinking any pattern, regardless of the complexity of its configuration. Problems regarding merging and disconnection of patterns during the process as well as the determination of the maximum number of steps necessary to obtain a single isolated element from a pattern, are reviewed and discussed. An analogy with a neural network description, in terms of McCulloch-Pitts "neurons" is presented.

Key Words and Phrases: counting binary patterns, shrinking, parallel processing, multiple connected pictures

CR Categories: 3.63

Introduction

The problem of counting binary patterns, i.e. patterns whose elements may assume only two values, 0 and 1, has been undertaken by several authors, including Izzo and Coles [1], Preston [2], Golay [3], and Weston [4]. All these authors used a sequential counting procedure, generally for biological applications. Weston's Numa-Rete is specially suited for counting objects even if they have intricate shapes or if some are contained in others; his approach is essentially hardware oriented. The other works [1–3] are based on the gradual erosion (shrinking) of the objects on a matrix, changing the 1's on their periphery into 0's, so that whenever an object is reduced to a single isolated 1-element, this is retained and counted. This procedure has been performed by computer programs. In this paper this second approach is followed, and finally a one-to-one correspondence between shrunk patterns and the series of natural numbers is obtained. Our method differs from that of the previous authors in employing the parallel processing procedure. Parallel processing of a picture denotes an operation by which the new value of each element is a local function of its neighbors, and all elements are processed independently and simultaneously. Conversely, in a sequential procedure, the new value of an element depends on the values of the surrounding elements, some of which have already been processed. During this work a computer program was written to simulate our parallel process with a sequential machine, but our aim is the implementation of hardware employing large-scale integration technology, as is developed in many research centers [5]. Although parallel computers were thought of many years ago [6, 7], the feasibility of these machines still depends on evaluating the amount of computation required to perform a given task [8].

The Algorithm

Our goal is to find a parallel shrinking algorithm for counting every connected (see below) set of 1-elements (object). The process will contract every object down to a single isolated point, and as soon as these points are obtained, they will be counted. Rosenfeld [9] has already investigated how, in the plane, a set S of points can be considered connected according to two different definitions which are simply related to the distance functions d_1 and d_2 [10]. In fact, a set S is said to be connected if, for any two considered elements in S, a path exists joining them through successive elements (i, j), (h, k) all in S, such that one of the following relations is satisfied.

(1) $d_1[(i, j); (h, k)] = |i - h| + |j - k| \leq 1$, and S is called d_1-connected;

(2) $d_2[(i, j); (h, k)] = \max(|i - h|, |j - k|) \leq 1$, and S is called d_2-connected.

The choice of these definitions of connectivity is not crucial; nevertheless, when we tackle line drawings containing segments oriented along any direction, a d_2-connectivity is more suitable since, for a correct use of the d_1-connectivity, oblique lines must be two elements thick. Sometimes this degree of thickness may be unacceptable.

Let us consider two-dimensional binary patterns on a matrix. Let the state of every element $a^*_{i,j}$ at step $n + 1$ be a local function of the states (0, 1) at step n of $a_{i,j}$ and of the elements of a 2×2 window including $a_{i,j}$.

$$a^*_{i,j} = \varphi(a_{i,j}\,;\, a_{i+1,j}\,;\, a_{i+1,j-1}\,;\, a_{i,j-1}).$$

φ may be expressed in terms of the Heaviside function $h(t)$ defined in the following way.

$$h(t) = 0 \quad \text{for} \quad t \leq 0,$$
$$h(t) = 1 \quad \text{for} \quad t > 0.$$

For the chosen window the algorithm will be

$$a^*_{i,j} = h[h(a_{i,j-1} + a_{i,j} + a_{i+1,i} - 1) + h(a_{i,j} + a_{i+1,j-1} - 1)]. \quad (1)$$

Shrinking will produce a compression of the pattern toward the top right. Any of the three remaining directions could have been considered and each is equally suitable for obtaining isolated elements. They are

$$a^*_{i,j} = h[h(a_{i,j} + a_{i,j+1} + a_{i+1,j} - 1) + h(a_{i,j} + a_{i+1,j+1} - 1)]$$

shrinks toward top left;

$$a^*_{i,j} = h[h(a_{i,j} + a_{i-1,j} + a_{i,j+1} - 1) + h(a_{i,j} + a_{i-1,j+1} - 1)]$$

shrinks toward bottom left;

$$a^*_{i,j} = h[h(a_{i,j} + a_{i,j-1} + a_{i-1,j} - 1) + h(a_{i,j} + a_{i-1,j-1} - 1)]$$

shrinks toward bottom right.

In order that an isolated element, i.e. one 1-element surrounded by eight 0-elements, may be extracted even when placed on the matrix border, an extra column of 0-elements must be added on the right and on the left of the original matrix, and an extra row must be added at the top and at the bottom.

Classical sequential shrinking algorithms process only 1-elements of the matrix, provided they are neither isolated nor interior and that their deletion does not locally disconnect, while 0-elements are disregarded by these algorithms.

Our parallel shrinking algorithm allows us to simplify considerably the previous algorithms, since all the matrix elements are involved in the process of shrinking. In fact, one is only required to look at a 2×2 window, while Rosenfeld has proved [9] that classical shrinking algorithms need at least a 3×3 window if they work in sequence, and a 5×5 window if they work in parallel. Rosenfeld's proof assumes that in any shrinking algorithm, 1-elements either remain such or become 0-elements at any step, but 0-elements never become 1-elements; whereas the present algorithm allows 0-elements to become 1-elements (i.e. in the configuration $\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}$).

As was previously mentioned, this algorithm has the property of being direction oriented: i.e. every object is compressed into one of the four corners of the rectangle circumscribing it. When, finally, only one point is obtained, it is extracted and counted before vanishing during the successive step [11]. This property is a necessity introduced in order to destroy the symmetry of certain configurations (see below) which we are unable to shrink in parallel unless we erase them completely.

1	0	1	1	0	1
0	1	1	1	1	0

Every $a_{i,j}$ gets connectivity information from the two by two window related to $a_{i,j}$ in the previous shrinking step.

Let φ, as defined in eq. (1), be the local operation which transforms the state of an element according to the states of the elements belonging to a two by two window, including the element itself.

THEOREM. *When φ is applied repeatedly in parallel on the matrix, a correct shrinking will be obtained; i.e.*

(1) only 1-elements that do not disconnect an object will be erased;

(2) 0-elements are not allowed to become 1 when this implies the merging of two or more objects.

PROOF. (1) We will analyze only those cases where a 1-element might become a 0-element; this happens only in the following case, where $a_{i,j} = 1$.

+	+	+	+
a	b	c	+
0	1	d	+
0	0	e	+

(+ elements may assume any value, 0 or 1.) We consider two possibilities:

(a) The $a_{i,j}$ element is an isolated one ($a = b = c = d = e = 0$): trivial case.

(b) $a_{i,j}$ belongs to such an object that would disconnect if the next state of $a_{i,j}$ were 0. This object will not be made by $a = b = c = d = e = 1$ but by $a = e = 1$ and $b = c = d = 0$. If such is the case, then when $a^*_{i,j} = 0$, simultaneously $b^* = 1$ and $d^* = 1$, so that d_2-connectivity is preserved.

This example reveals that the object is compressed toward the top right.

(2) Let us now consider those cases in which $a^*_{i,j} = 1$ when $a_{i,j} = 0$:

f	g	h	i		f	g	h	i
a	b	c	k		a	b	c	k
1	0	d	l	and	1	0	d	l
0	1	e	m		1	1	e	m

We see that these cases are equivalent from a d_2-connectivity point of view; therefore we will consider only the first of them.

Let us look at the matrix and consider in which position another object (not connected to the given one, which is constituted by $a_{i,j-1}$ and $a_{i+1,j}$) can be placed so that the transformation of $a_{i,j} = 0$ into $a^*_{i,j} = 1$ would merge them. This object must have at least $c = 1$; g, h, i, k, and l might have any value, but $a = b = d = e = 0$. According to the previous considerations, the matrix may be arranged as follows.

+	1	+	+
0	0	1	+
1	0	0	1
0	1	0	+

If we apply (1) to the matrix, we see that $a^*_{i,j} = 1$ but $c^* = 0$; therefore, both objects will still remain disconnected.

Shrinking Procedure

THEOREM. *When φ is applied repeatedly in parallel on the matrix, a single isolated element will be obtained for each object present on it.*

For our specific algorithm (1) let us consider two cases: (a) singly connected object; (b) multiply connected object.

a	1	1	e	h	k
0	1	1	f	i	l
0	1	1	g	j	m
b	1	1	1	1	1
0	c	1	1	1	1
0	0	d	1	0	0

Initial values:
$a = b = c = d = 1$
$e = f = g = h = i = j = k = l = m = 0$

Case (a). Let us consider the 1-elements of the object as belonging to three subsets: those lying on the bottom and left borders, where a border element is a 1-element with at least a 0-element at a distance $d_2 = 1$, (subset L); those lying on the top and right borders (subset M); and the remaining ones (subset N).

Every $a_{i,j}$ in subset L will have at least one 0-element in its own window; therefore, $a^*_{i,j} = 0$ or $a^*_{i,j} = 1$. On the other hand, let us consider the subset of 0-elements adjacent (i.e. at a distance $d_2 = 1$) to the elements belonging to the subset L. We observe that none of these 0-elements will be transformed into 1-elements, so that the total number of elements belonging to the subset L will not increase; but, since there is at least one $a_{i,j}$ in L for which $a^*_{i,j} = 0$ (look at a, b, c, d on the explanatory matrix above), then the total number of elements in L will decrease.

As for the subset M, if we consider 0-elements on the top and right borders of the object, these might remain 0 or become 1-elements; this will happen whenever the window containing it is like the one shown:

1	0
1	1

This means that the elements of subset M will eventually increase until a full corner configuration is reached ($e = f = g = h = i = j = k = l = m = 1$ on the explanatory matrix). The elements belonging to subset N, inside the object, will remain in their state.

Now at each shrinking step some of the 1-elements of L will disappear, provoking an erosion of the object, and some 0-elements adjacent to the 1-elements of M will become 1-elements until a full corner configuration is reached. At this point, each iteration of the process will decrease the total number of 1-elements until only the top right corner 1-element (isolated element) is obtained.

Case (b). The object is multiply connected and we can again divide the elements of the picture into three subsets: subsets L and M are as defined in case (a), while subset N contains all the remaining 1-elements belonging to the object and those 0-elements lying on its inside (elements belonging to holes).

As for the subsets L and M, their elements will be processed like those in the corresponding subsets of case (a). Conversely, the elements belonging to the subset N will gradually all become 1-elements, while the process is iterated. In fact, whatever the transformed states of the elements belonging to subset N may be, the 1-elements of subset L will provoke compression and hole filling in such a way that at a certain shrinking step the behavior of subset N will be the same as that of its corresponding subset of case (a).

We have seen that during the shrinking process, the top rightmost borders of the object move until a full corner configuration (as described before) is obtained.

The corner element (c.e.) will be the isolated element representing the object. Every 1-element belonging to the object will have a certain d_1 distance with respect to the c.e. One or more 1-elements will have a maximum d_1 distance D_1. These 1-elements will obviously belong to subset L.

COROLLARY. *The maximum number of steps necessary for shrinking a given object to an isolated 1-element is equal to the d_1 distance between the top rightmost corner of the rectangle circumscribing the object and its 1-element most distant from it.*

PROOF. Every shrinking step will delete only 1-elements belonging to the left and bottom borders, and therefore $D_1^* = D_1 - 1$ where D_1^* represents the transformed d_1 maximum after one shrinking step. Only when the object is reduced to the c.e., $D_1^* = 0$, and therefore the maximum number of steps coincides with D_1.

This is a maximum number because whenever special patterns of the following kind appear, a smaller number of steps is required.

0 0 1
0 1 0
1 0 0

$D_1 = 4$, but only two steps are needed.

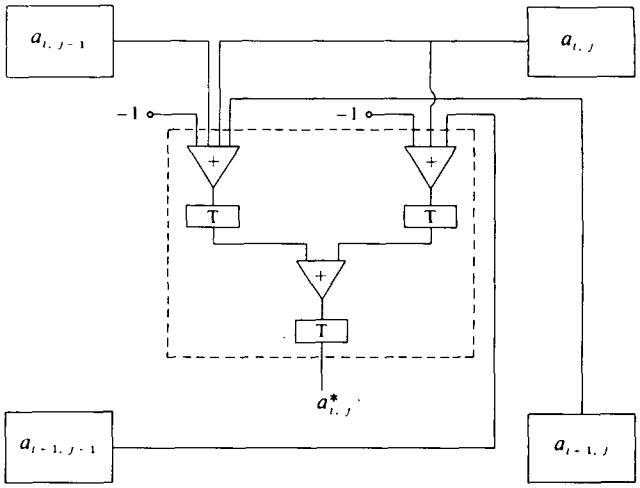
Conclusions

We have considered an algorithm for counting binary patterns using a shrinking procedure. This algorithm differs from others in two respects: (1) it is essentially a parallel processing algorithm; (2) the transformed state of every element on the matrix is a function only of the state of the elements of a two by two window including the element itself.

The algorithm was implemented using integrated circuitry with an eight by eight input matrix [11], but hardware employing large-scale integrated circuitry is well under way for a specific counting device. A simulation of the algorithm was performed on a sequential computer, but since the algorithm is parallel in nature, a storage matrix was needed. It was seen, as expected, that correct counting of binary patterns was obtained, regardless of the complexity of their configuration, even when some patterns were contained in others.

Furthermore, we can also think in terms of networks of "neurons" such as those schematized by McCulloch and Pitts [12, 13]. We might then consider a network made up of two parts: one representing the input matrix (on which patterns are given) and another which is the storage matrix (on which transformed elements are sequentially written). A "neuron" could be on (1) or off (0), according to the state of the element it represents; the coupling coefficients and the thresholds would be fixed. Figure 1 is a simple block diagram for such a neuron. The information switches from one half of the network to the other half, obtaining shrinking as a final goal. Such a network would be general in nature but specific in scope and may be thought of as a parallel computing system on its own.

Fig. 1. "Neuron" that computes the shrinking algorithm.



Acknowledgment. I thank C. Arcelli for his invaluable help in discussing the revised version of this paper.

Received February 1970, revised February 1971

References

1. Izzo, N.F., and Coles, W. Blood cell scanner identifies rare cells. *Electronics 35*, 27 (Apr. 1962), 52–57.
2. Preston, K. Jr. The CELLSCAN system—A leucocyte pattern analyzer. Proc. Western Joint Comput. Conf., Vol. 19, May 1961, Spartan Books, New York, p. 175.
3. Perkin-Elmer Engineering Rep. 5489A (unpublished).
4. Weston, P. Photocell field counts random objects. *Electronics 34*, 22 (Sept. 1961), 46–47.
5. Heath, F.G. Large scale integration in electronics. *Scientific American 222* (Feb. 1970), 22–31.
6. Unger, S.H. Pattern detection and recognition. Proc. IRE, Vol. 47, 1959, pp. 1737–1752.
7. McCormick, Bruce H. The Illinois pattern recognition computer, Illiac III. *IEEE Trans. Electronic Computers*, (Dec. 1963), 791–813.
8. Minsky, M., and Papert, S. *Perceptron, An Introduction to Computational Geometry*. The MIT. Press, Cambridge, Mass., 1969.
9. Rosenfeld, A. Connectivity in Digital Pictures, *J. ACM 17*, 1, (Jan. 1970), 146–160.
10. Rosenfeld, A., and Pfaltz, J.L. Distance functions on digital pictures. *Pattern Recognition. 1*, 1 (July 1968), 33–61.
11. Levialdi, S. Parallel counting of binary patterns, *Electronics Letters 6*, (1970) 798–800.
12. McCulloch, W. S., and Pitts, W. A logical calculus of ideas immanent in nervous activity. *Bull. Math. Biophys.* 5 (1943), 115–137.
13. Caianiello, E.R. Outline of a theory of thought processes and thinking machines, *J. Theoret. Biol. 1* (1961), 204–235.