# Experiments on the Decomposition of Arbitrarily Shaped Binary Morphological Structuring Elements [*]

Giovanni Anelli, Alberto Broggi[**], and Giulio Destri

Dipartimento di Ingegneria dell'Informazione,
Università di Parma, I-43100 Parma, Italy

**Abstract.** The decomposition of binary structuring elements is a key problem in morphological image processing. So far only the decomposition of convex structuring elements and of specific subsets of non-convex ones have been proposed in the literature. This work presents the results of a new approach, based on a Genetic Algorithm, in which no constraints are imposed on the shape of the initial structuring element, nor assumptions are made on the elementary factors, which are chosen from a given set.

## 1 Introduction

*Mathematical Morphology* [8,12,6] concerns the study of shape using the tools of set theory. Mathematical morphology has been extensively used in low-level image processing and analysis applications, since it allows to filter and/or enhance only some characteristics of objects, depending on their morphological shape.

Within the mathematical morphology framework, a binary image $A$ is defined as a subset of the two-dimensional Euclidean space $E^2$ $(Z \times Z)$:

$$A = \{a = (a_i, a_j) \mid a_i, a_j \in Z\} \tag{1}$$

In [6], monadic transforms acting on a generic image $A$ (*complement*, *reflection*, and *translation*) and dyadic operators between sets (*dilation*, *erosion*, *opening*, and *closing*) are defined. This work will focus only on *dilations*,

$$A \oplus B \triangleq \{x \in E^2 \mid x = a + b, \text{ for some } a \in A, b \in B\} \tag{2}$$

where $A$ represents the image to be processed, and $B$ is called *Structuring Element* (SE), i.e. another subset of $E^2$ whose shape parameterizes each operation.

A SE $B$ is said to be *convex with respect to a given set of morphological operations (e.g. dilation) with a given set of SEs (factors)* $\{F_i, i = 1, ..., n\}$ if it can be expressed as a chain of dilations of the $F_i$ elements:

$$B = F_{k_1} \oplus F_{k_2} \oplus F_{k_3} \oplus \ldots \oplus F_{k_m}, \quad \text{with } k_j \in [1, n], \quad \text{for } j = 1, ..., m \tag{3}$$

[*] This work has been partially supported by the 'Progetto Finalizzato Trasporti 2' of the Italian CNR.

[**] E-Mail: broggi@CE.UniPR.IT

Otherwise $B$ is said to be *non-convex with respect to the same set of SEs*, and thus it can only be expressed as a chain of boolean operations (e.g. unions and/or intersections) between convex elements (*partitions*):

$$B = B_1 \odot B_2 \odot B_3 \odot ... \odot B_z, \tag{4}$$

where $\odot$ represents any boolean operation (such as unions $\cup$, intersections $\cap$,...) and $B_i$ are convex elements that can be expressed as chains of dilations, as shown by equation (3).

The decomposition of a binary SE into a chain of operations involving only elementary factors is a key problem [1]. So far, only *deterministic* solutions have been analyzed and proposed in the literature [3,10,11,13], each relying on different assumptions (such as convex SEs, specific sets of elementary operators, etc.); on the other hand the optimal decomposition (with respect to a given set of optimality criteria) of non-convex generic SEs with a *deterministic* approach is still an open problem.

This paper addresses this problem utilizing a *stochastic* approach, based on Evolution Programs: starting from a population of potential solutions (individuals), an iterative process modifies the existing individuals and/or creates new ones in accordance to some given functions applied randomly. The best solutions in the population tend to replace the others, and, after a sufficient number of iterations, the algorithm tends to converge toward the optimal solution.

In particular, two are the purposes of this work:

- to apply the results to a real-world case in which the decomposition of SEs is a basic programming technique: the determination of the optimal decomposition with respect to the instruction set of the massively parallel architecture PAPRICA, in order to execute operations based on complex SEs;
- to determine a performance index for the instruction set of a generic massively parallel cellular system dedicated to morphological tasks.

Next section introduces the problem of SE decomposition; section 3 briefly summarizes the approach, and section 4 discusses the results of the stochastic decomposition. Section 5 concludes the paper with some remarks and future research directions.

## 2 Structuring Element Decomposition on SIMD Systems

This section addresses the problem of the optimal decomposition of a complex SE on *SIMD cellular systems*, whose operations are based on a neighborhood smaller than the size of the SE. In the following examples, a *dilation* between a generic image $A$ and a complex SE $B$ is considered; due to the properties of unions and intersections discussed in [6], namely

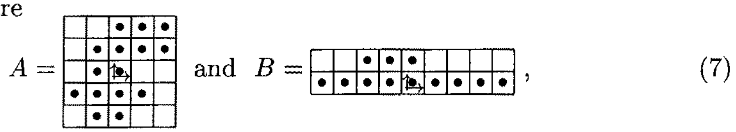
$$A \oplus (B_1 \cup B_2) = (A \oplus B_1) \cup (A \oplus B_2) \tag{5}$$

$$A \oplus (B_3 \cap B_4) \subseteq (A \oplus B_3) \cap (A \oplus B_4), \tag{6}$$

in the following we prefer to express a non-convex SE as a chain of *unions* of convex SEs, as in equation (5), instead of using *intersections* or other boolean operations.

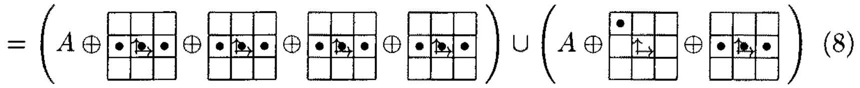
In cellular systems the set of all possible operations (known as *Instruction Set*, IS) is generally based on 3 × 3 SEs. Thus the main constraint that must be considered in the decomposition of complex SEs is that each elementary operation must belong to the Instruction Set.

Assuming a system capable of performing horizontal and vertical dilations, and translations in the 8 main directions, the SE $B$ of the following dilation $R = A \oplus B$, where

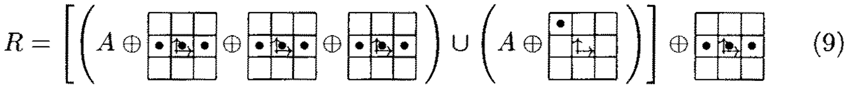


is non-convex with respect to the IS of the system. It can be expressed as a union of convex sets, for example:

$$R = A \oplus B = A \oplus (C_1 \cup C_2) = (A \oplus C_1) \cup (A \oplus C_2) =$$



Eq. (8) contains 6 elementary dilations and 1 logical union. Using the chain rule property, $R = A \oplus B$ can be expressed with a two-level solution as:



This solution requires only 5 dilations and 1 logical union.

## 2.1 Optimality criteria

The decomposition of a SE can be aimed to many different goals, such as:

- the minimization of the number of decomposing sets (to reduce the number of dilations);
- the minimization of the total number of elements in the decomposing sets (to reduce the size of the data structures and thus also the memory requirements in serial systems);
- the minimization of the total number of computations (for speed-up reasons);
- the possibility to implement complex morphological operations on cellular systems whose IS is based on simple, elementary operations (to overcome the problem caused by the simple interconnection topology that limits the size of possible SEs);
- or even the determination of factors with a given shape (to ease the recognition of 2D objects).

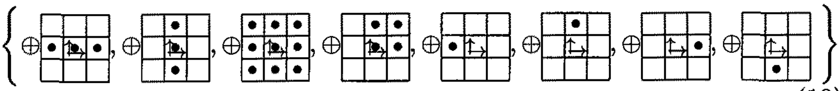
The optimality criterion addressed in this work is the reduction of the computational complexity of the processing, namely the minimization of the number of elementary operations required to perform morphological processings based on large and complex SEs.

### 2.2 A case study: PAPRICA system

PAPRICA [4] is a special-purpose SIMD massively parallel coprocessor designed to be installed on a moving vehicle for vision-based obstacle detection and lane keeping tasks [2]. Special care has been devoted to the design of the IS of the machine since it affects directly the system performance and effectiveness: the minimization of the number of elementary instructions that must be combined together to synthesize an operation based on a complex SE is a task that extends far beyond the optimization of a specific algorithm, involving also the definition of a sufficiently general IS.

Beside logical operations, PAPRICA IS is composed of the following morphological operations:

$\mathrm{IS_P} =$



(10)

## 3 Implementation of the Genetic Approach

Genetic Algorithms (GAs) are optimization algorithms based on a stochastic search [7], widely used in various fields [5]. They use ideas taken from the biology mechanism to drive the search toward an optimal, or nearly optimal solution: the terminology used in GAs has thus been imported from biology. GAs operate on a *population* of potential solutions for the considered problem (*individuals*) by means of *genetic operators*. Each individual contains a *Genome* or *Chromosome*, that is composed by a set of *Genes*, representing the function parameters and by a *Fitness value*, the result of the *evaluation function*, measuring the “goodness” of the solution encoded in this individual. The genetic search is driven by the *fitness* values of the individuals: each individual must be evaluated to give some quantitative measure of its *fitness*, that is the “goodness” of the solution it represents. At each iteration (*generation*) the fitness evaluation is performed on all individuals. Then, at the following iteration, a new population of potential solutions (*Offspring*) is generated, starting from the individuals with the highest fitness, and replacing, completely or partially, the previous generation.

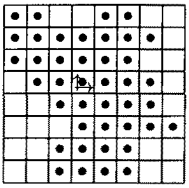
The genetic operators used to generate new individuals are subdivided into two main categories: *unary* operators, creating new individuals and replacing the existing ones with a modified version of them (e.g. *mutation*,introduction of random changes of genes), and *binary* operators, creating new individuals through the combination of data coming from two individuals (e.g. *crossover*, exchange of genetic material between two individuals). Each iteration step is called *generation*.

The study of GAs led to the more general Generalized GAs or *Evolution Programs* (EPs) [9]. In “standard” GAs an individual is represented by a fixed-length binary string, encoding the parameter set, which corresponds to the solution it represents; the genetic operators act on these binary codes. In EPs, individuals are represented as generalized data structures without the fixed-length constraint. The programmer can choose the most appropriate data structure with respect to the specific problem, for example operating in the same parameters space of the application. In addition, ad-hoc operators are defined to act on these data structures.

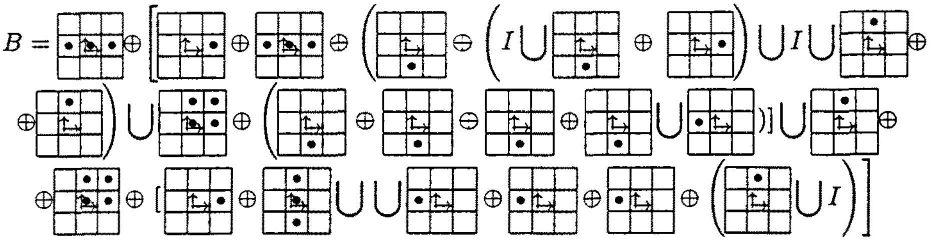
When the genetic approach is applied to the SE decomposition problem, its intrinsic nature, that is the *varying number* of elementary items forming a solution, does not allow to know a priori the size of a generic solution, that is the length of the coding of a generic individual. The data structure representing the individual must explicitly encode both the number and the shape of each single elementary operation composing the solution. Moreover, the coding must allow also a quick and easy evaluation phase. An ad-hoc EP has thus been developed, exploiting a method similar to the solution of the bin-packing problem.

## 4 Analysis of the Results

Let us now consider the decomposition of the following non-convex SE $B$, whose optimal decomposition is definitely non-trivial:

$B =$  (11)

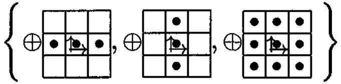
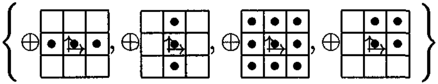
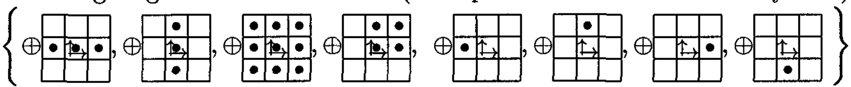
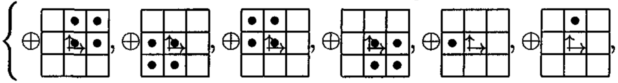
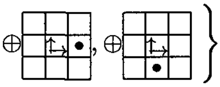
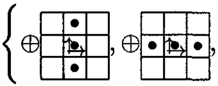
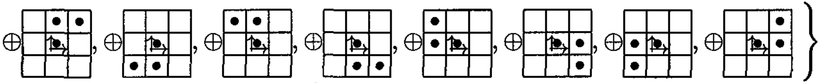
After 300 generations on a population of 2000 individuals, taking less than one hour of processing time on a Sun Sparc station 20, considering PAPRICA IS shown in (10), the stochastic decomposition led to 50 elementary dilations and 8 logical unions. The corresponding two-level solution comprises 22 elementary dilations and 8 logical unions, as shown in the following:



where $I$ is the identity image. This decomposition allows the direct translation of the solution into PAPRICA Assembly code (more details can be found in [4]):

```
L2 = HEXP(L1)      L10 = WMOV(L9)     L18 = SMOV(L17)      L26 = (L15) | L25
L3 = NMOV(L1)      L11 = EMOV(L4)     L19 = SMOV(L18)      L27 = (L19) | L26
L4 = NEEXP(L3)     L12 = VEXP(L11)    L20 = WMOV(L7)       L28 = (L20) | L27
L5 = HEXP(L2)      L13 = SMOV(L6)     L21 = NMOV(L10)      L29 = (L21) | L28
L6 = EMOV(L5)      L14 = NMOV(L6)     L22 = SMOV(L13)      L30 = (L10) | L29
L7 = NEEXP(L2)     L15 = NMOV(L14)    L23 = EMOV(L22)      L31 = (L12) | L30
L8 = WMOV(L4)      L16 = SMOV(L7)     L24 = (L13) | L23
L9 = WMOV(L8)      L17 = SMOV(L16)    L25 = (L6) | L24
```

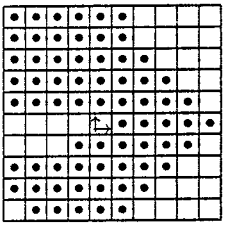
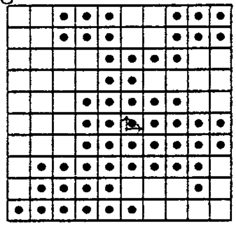
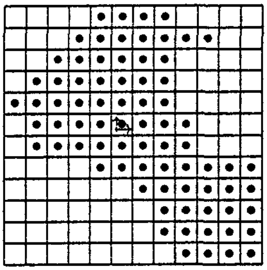
### 4.1 Instruction set evaluation

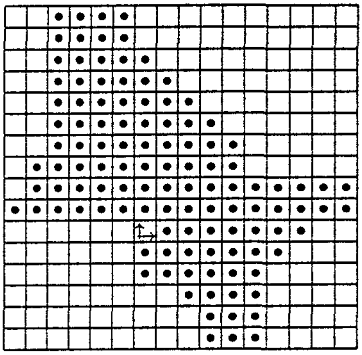
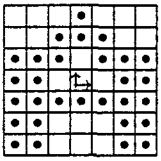
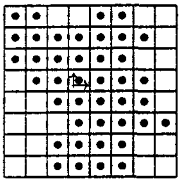
Different ISs have been evaluated for the decomposition of the same set of SEs, starting from the simplest $IS_1$ = , extending it to a RISC-oriented non-symmetric set $IS_2$ = , including single direction dilations (as implemented on PAPRICA system) $IS_3$ = , replacing the first 4 elements with symmetrical ones, according to a CISC-oriented implementation $IS_4$ =  , up to a completely unusual set $IS_5$ =  .

The average number of operations $N$ required to synthesize a sufficiently large set of different SEs (normalized with respect to the number of operations $N(IS_1)$ required by $IS_1$) is given in table 1, showing that the solution adopted for PAPRICA system represents a good trade-off between the complexity of the IS (in terms of number of factors) and its potentialities.

In the following results are compared in two different cases: with different SE (a) size and (b) shape.

**a)** The size of the 6 following SEs

$F3 =$  $F4 =$  $F5 =$ 

$F6 =$  $F1 =$  $F2 =$  varies from 7×7 up to 16×16; the single-level decomposition has been performed with respect to the 5 different instruction sets presented above; the results are shown in table 2.

| Instruction Set | $IS_1$ | $IS_2$ | $IS_3$ | $IS_4$ | $IS_5$ |
|---|---|---|---|---|---|
| $N/N(IS_1)$ | 1 | 0.93 | 0.82 | 0.84 | 0.71 |

**Table 1.** Average IS performance

| SE | $IS_1$ | $IS_2$ | $IS_3$ | $IS_4$ | $IS_5$ |
|---|---|---|---|---|---|
| F1 | 24(16/8) | 24(16/8) | 20(15/5) | 21(16/5) | 23(16/7) |
| F2 | 35(24/11) | 30(22/8) | 26(20/6) | 26(20/6) | 19(15/4) |
| F3 | 32(23/9) | 32(24/8) | 29(22/7) | 30(23/7) | 23(19/4) |
| F4 | 52(38/14) | 47(35/12) | 39(31/8) | 41(33/8) | 43(34/9) |
| F5 | 42(31/11) | 38(30/8) | 36(29/7) | 38(31/7) | 29(24/5) |
| F6 | 62(47/15) | 55(42/13) | – | – | – |

**Table 2.** Results of the decomposition of SEs $F1, F2, ..., F6$. Only two of the decompositions of $F6$ could be performed due to the extremely large memory requirements. The first number represents the total number of instructions; between brackets the number of morphological and boolean operations, respectively.

**b)** Table 3 shows the results of the decomposition of the following three 7 × 7 SEs (used in the search for planes in aerial images):

$$G1 = \begin{array}{|c|c|c|c|c|c|c|}\hline & \bullet & \bullet & \bullet & & & \\ \hline & \bullet & \bullet & \bullet & \bullet & & \\ \hline & \bullet & \bullet & \bullet & \bullet & \bullet & \\ \hline & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ \hline & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ \hline & \bullet & \bullet & \bullet & & \bullet & \bullet \\ \hline & \bullet & \bullet & \bullet & & & \bullet \\ \hline \end{array} \quad G2 = \begin{array}{|c|c|c|c|c|c|c|}\hline & & & & & \bullet & \bullet \\ \hline & & & & \bullet & \bullet & \bullet \\ \hline & & & \bullet & \bullet & \bullet & \bullet \\ \hline & & \bullet & \bullet & \bullet & \bullet & \\ \hline & \bullet & \bullet & \bullet & \bullet & & \\ \hline & \bullet & \bullet & \bullet & \bullet & & \\ \hline & \bullet & \bullet & \bullet & \bullet & & \\ \hline \end{array} \quad G3 = \begin{array}{|c|c|c|c|c|c|c|}\hline \bullet & & & & & & \bullet \\ \hline \bullet & \bullet & & & & \bullet & \bullet \\ \hline \bullet & \bullet & \bullet & & \bullet & \bullet & \bullet \\ \hline & \bullet & \bullet & \bullet & \bullet & \bullet & \\ \hline & & \bullet & \bullet & \bullet & & \\ \hline & & \bullet & \bullet & \bullet & & \\ \hline & & \bullet & \bullet & \bullet & & \\ \hline \end{array} \tag{12}$$

is shown in table 3.

| SE | $IS_1$ | $IS_2$ | $IS_3$ | $IS_4$ | $IS_5$ |
|---|---|---|---|---|---|
| G1 | 16(11/5) | 15(10/5) | 13(10/3) | 14(11/3) | 11(9/2) |
| G2 | 17(10/7) | 16(9/7) | 13(9/4) | 13(9/4) | 7(6/1) |
| G3 | 21(14/7) | 21(14/7) | 20(14/6) | 20(14/6) | 15(11/4) |

**Table 3.** Results of the decomposition of SEs $G1, G2, G3$.

The example shows that, as a general rule, the larger the number of operations in the IS, the shorter the decomposition; but it also shows that established a fixed number of factors in the IS, a risc-oriented solution provides better results.

## 5 Conclusion

This paper presented the results of the use of a stochastic approach to the decomposition of arbitrarily shaped binary morphological structuring elements into chains of elementary factors. The application of this technique to *convex* SEs leads to the optimal decomposition discussed in the literature [1]; in addition, this paper addressed the decomposition of generic *non-convex* SEs.

In particular, two are the main results reported in this work:

- first, it is possible to derive automatically the PAPRICA assembly program from the result of the decomposition; when the optimality criterion used in the decomposition is the minimization of the number of elementary morphological operations, this corresponds to the determination of the PAPRICA program requiring the minimum computational time.
- second, the decomposition of a large number of SEs was used to derive a performance index for different ISs; in particular the one implemented on the PAPRICA system demonstrated to be a good trade-off between performance and complexity (intended as the number of elements in the IS).

Due to the extremely high computational load and to the large memory requirements needed by the iterative approach, the genetic engine is now being ported to the MPI parallel environment in order to speed-up the processing and to allow the decomposition of very large SEs.

A graphical interface is also under development to ease the definition of both the initial SE and the IS, as well as the introduction of parameters. The interface, based on Java, will allow remote users to run their own decompositions using the new Web technology. The first release of the complete tool running on Unix systems under the MPI environment will be shortly available as Public Domain software.

## References

1. G. Anelli, A. Broggi, and G. Destri. Decomposition of Arbitrarily Shaped Binary Morphological Structuring Elements using Genetic Algorithms. *IEEE Trans PAMI*, 1997. In press.
2. M. Bertozzi and A. Broggi. GOLD: a Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection. *IEEE Trans Image Processing*, 1997. In press.
3. A. Broggi. Speeding-up Mathematical Morphology Computations with Special-Purpose Array Processors. In *Procs of the 27th HICSS*, vol I, pages 321–330, 1994.
4. A. Broggi, G. Conte, F. Gregoretti, C. Sansoè, R. Passerone, and L. M. Reyneri. Design and Implementation of the PAPRICA Parallel Architecture. *The Journal of VLSI Signal Processing*, 1997. In press.
5. D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison Wesley, Readings, MA, 1989.
6. R. M. Haralick, S. R. Sternberg, and X. Zhuang. Image Analysis Using Mathematical Morphology. *IEEE Trans PAMI*, 9(4):532–550, 1987.
7. J. Holland. *Adaption in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, 1975.
8. G. Matheron. *Random Sets and Integral Geometry.* John Wiley, New York, 1975.
9. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, Berlin, 1992.
10. H. Park and R. T. Chin. Optimal Decomposition of Convex Structuring Elements for a 4-Connected Parallel Array Processor. *IEEE Trans PAMI*, 16(3), March 1994.
11. H. Park and R. T. Chin. Decomposition of Arbitrarily Shaped Morphological Structuring Elements. *IEEE Trans PAMI*, 17(1), January 1995.
12. J. Serra. *Image Analysis and Mathematical Morphology.* Academic Press, 1982.
13. X. Zhuang and R. M. Haralick. Morphological structuring element decomposition. *Computer Vision, Graphics, and Image Processing*, 35:370–382, September 1986.