A parallel Homological Spanning Forest framework for 2D topological image analysis

Fernando Diaz-del-Rio [a], Pedro Real [b], Darian M. Onchis [c, d]

[a] Computer Architecture and Technology Department, University of Seville, Avda. Reina Mercedes s/n, Seville 41012, Spain
[b] Institute of Mathematics, University of Seville, Avda. Reina Mercedes s/n, Seville 41012, Spain
[c] Faculty of Mathematics, University of Vienna, Oskar-Morgenstern-Platz 1, Wien A-1090, Austria
[d] Department of Electrical Engineering and Informatics, Eftimie Murgu University, Traian Vuia Square 1–4, Resita, Romania

Keywords:
Computational algebraic topology
2D digital image
Primal–dual abstract cell complex
Homological Spanning Forest
Topological analysis
Parallel algorithm

A B S T R A C T

In [14], a topologically consistent framework to support parallel topological analysis and recognition for 2*D* digital objects was introduced. Based on this theoretical work, we focus on the problem of finding efficient algorithmic solutions for topological interrogation of a 2*D* digital object of interest *D* of a pre-segmented digital image *I*, using 4-adjacency between pixels of *D*. In order to maximize the degree of parallelization of the topological processes, we use as many elementary unit processing as pixels the image *I* has. The mathematical model underlying this framework is an appropriate extension of the classical concept of abstract cell complex: a primal–dual abstract cell complex (pACC for short). This versatile data structure encompasses the notion of Homological Spanning Forest fostered in [14,15]. Starting from a symmetric pACC associated with *I*, the modus operandi is to construct via combinatorial operations another asymmetric one presenting the maximal number of non-null primal elementary interactions between the cells of *D*. The fundamental topological tools have been transformed so as to promote an efficient parallel implementation in any parallel-oriented architecture (GPUs, multi-threaded computers, SIMD kernels and so on). A software prototype modeling such a parallel framework is built.

1. Introduction

This paper is concerned with the problem of developing a topologically-consistent framework for efficient parallel topological analysis of discrete objects in 2D digital imagery. The topological consistency proof of such systems is provided in most of the cases by means of a mathematical model of digital images and objects, under which all theoretical formulae related to topology are true and there is no room for paradoxes. Moreover, this framework must substantially simplify the algorithmic design of an advanced *topological calculus and recognition* of the objects of interest. We aim to achieve parallel architectures compatible with this framework and to reduce drastically the time complexity in topological computations. In order to avoid segmentation issues and noise which are common to mathematically ill-posed problems ubiquitous in the area of Digital Imagery, the input data are 2-dimensional integer-valued matrices associated with a binary or gray-level pre-segmented 2D digital image *I*. Our interest here is to design and to implement a parallel framework providing efficient and fast algorithmic answer to any topological interrogation for a region of interest (ROI for short) *D* of *I*, using 4-adjacency between pixels of *D*.

Roughly speaking, topology helps to understand the different "degrees of connectivity" a geometric object has. To deal with topological isomorphisms or homeomorphisms between continuous geometric objects is a very hard task and discretization strategies, such as triangulations, are employed for reducing the computational complexity of the topological interrogation. Within a semi-continuous context, geometric subdivided objects are commonly represented by cell or CW-complexes (for example, [16]). Finally, within a purely discrete level, combinatorial versions of CW-complexes, called *abstract cell complexes* (ACC, for short), can be used for a correct algorithmic development. They are formed of basic elements (representing the cells using topological coordinates) of different dimension together with a bounding function describing the combinatorial relationship "to be in the boundary of". Different definitions of ACCs can be found in the literature (see [11] for a thorough survey).

Concerning the computability of topological features and invariants, there are two main ways for computing *n*-dimensional

"(co)holes" measuring the lack of connectivity: (co)homology and (co)homotopy. Homology considers the notion of hole in linear algebra terms and homotopy in purely combinatorial terms. All the previous descriptions of ACCs are convenient for codifying chain complexes (algebraic versions of cell complexes) and designing homological computational techniques. Nevertheless, homotopy computation is much more harder in general than homology computation and cannot be appropriately developed in a parallel framework exclusively based on this ACC coding. An exception to this is given by the Euler number (see, for example [2]) that can be computed exclusively using local information on pixels.

We use as model of our parallel combinatorial framework an extension of the classical ACC notion called *primal–dual abstract cell complex*, in the sense that two bounding functions are employed for specifying the connectivity of the structure. This notion encompasses the concept of HSF developed in [14,15] which ensures the reliability of topological interrogation both at homology and homotopy levels. In fact, an HSF of a cell complex D can be seen as an asymmetric pACC strongly "connected" (in combinatorial terms) to a fully symmetric pACC defining D. From this topological model in 2D digital ambiance, we can reach homotopy-type features and characteristics like topological trees or thinning.

There are numerous contributions in the literature (see, for example, [8–10,12,13,17,19,22,24]) dealing with parallel algorithms computing a concrete topological invariant or feature (for example, Euler number, connected component labeling, simple points and thinning algorithms, Betti numbers, persistent homology, topological trees,....) for a region of a 2D digital image. A much more reduced quantity of papers (see, for example, [1,4,14]) propose a theoretical topologically-consistent framework in this digital context. To our knowledge, the present paper is the first one implementing such a parallel framework for advanced topology computation.

The combinatorial technique used here for constructing asymmetric pACCs encoding faithful topological information takes some inspiration from methods developed in Simple Homotopy Theory [25,26], Discrete Morse Theory [5,6], Effective Homology [23] and, mainly Algebraic-Topological models [7,14,15,20].

Summing up, the primary contributions of this work is to design and implement a purely combinatorial algorithm for constructing topological HSF models of two-dimensional digital objects in a parallel architecture context.

2. Topological rationale

The different steps of any topological processing in our framework are in order: (a) Input data; (b) Extraction of the ROI; (c) Elementary sequential or parallel step; (d) Output: Topological model of the ROI. Following this pipeline, we gradually introduce the main mathematical notion involved in the present generic framework: a pACC. The computational techniques for generating new special pACCs employed in stages (c) and (d) will be detailed in the next two sections.

- **(a) Input data:** The pair (I, D). The 2D digital image $I: \{1,\dots,m\}x\{1,\dots,n\} \to \{0,1,\dots,2^k-1\}$ is represented by a $m \times n$ $(m,n,k \in \mathbb{N})$ integer-valued matrix. The digital object D, called region-of-interest (or ROI, for short), is formed by set of pixels (given by their "coordinates" row-column) of I. For example, D can be the set $I^{-1}(r)$, for some $r \in \{0,1,\dots,n-1\}$.
- **(b) Extraction of the ROI:** From I, we extract the ROI D by means of new digital image I_D of the same dimension than I. The set of black pixels (numbered by 1's) of I_D is exactly D.
- **(c) Elementary sequential or parallel step: generation of topological pACCs:** In this phase, we compute two kind of pACCs: (a) symmetric ones, modeling D and I in terms of cell complexes; (b) asymmetric ones, starting from to the previous symmetric pACCs, called Morse Spanning Forest. We distinguish three parts:

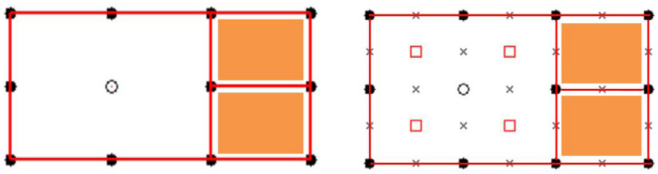

Fig. 1. Two combinatorial scenarios: (Left) *ACC* associated with a digital object D formed by black pixels within a 3×4 digital image I; (Right) $pACC(D) \subset pACC(I)$.

(c.1) **Embedding in regular subdivided scenario**. Our strategy for processing digital objects within a digital image is based on constructing topological models within an appropriate combinatorial "ambiance" context. This scenario in which we need to embed the digital image I_D is a pACC intimately associated with the contractible set of cells denoted by $Cell(I_D)$. $Cell(I_D)$ only depends on the dimensions of I_D and can be constructed in a straightforward way: 0-cells are the pixels of I_D (black or whites), 1-cells are given by the set of two 4-adjacent pixels (horizontal or vertical) and 2-cells are given by sets of four mutually 4-adjacent pixels. Thus, a dimension function dim: $Cell(I_D) \to \{0, 1, 2\}$ is well-defined in this way. One cell c' is in the boundary (resp. in the coboundary) of another c'' if the set of pixels of c' (resp. c'') is included in the set of pixels of c'' (resp. c'). In an analogous way, we can construct the set of cells $Cell(D)$, in which only the black pixels of I_D are involved.

In Fig. 1, two possible combinatorial scenarios for topology computation are shown: that of classical ACC, good enough to efficiently compute homology features and characteristics (left); and that of pACC (right). The last one will make possible to give us fast answers to homotopy interrogation problems. Although the second option seems to introduce a higher complexity, it can be reduced by means of a careful local analysis in the huge ambiance space that it builds.

Now, it is time to define the notion of pACC within a general n-dimensional setting. ***A finite primal–dual abstract cell complex (pACC for short)*** $\mathsf{C} = (C, B^p, B^d, dim_p, dim_d)$ is composed of:

- $C \bigcup \{\emptyset\}$, where C is a finite set of cells.
- Two dimension functions: *(primal dimension)* $dim_p : C \to \{0,1,2,\dots,k_p\}$ and *(dual dimension)* $dim_d : C \to \{0,1,2,\dots,k_d\}$, where $k_p, k_d \in \mathbb{N} \cup \{0\}$. The primal and dual dimension of the empty set $\emptyset$ is -1. The set C_i^p (resp. C_i^d) is the set of cells such that their primal (resp. dual) dimension is i.
- Two bounding relations: (primal bounding relation) a graded function $B_i^p : C_i^p \times C_{i+1}^p \to \mathbb{N} \cup \{0\}$ $(\forall 0 \le i \le k_p - 1)$ and (dual bounding relation) a graded function $B_i^d : C_i^d \times C_{i+1}^d \to \mathbb{N} \cup \{0\}$, $\forall 0 \le i \le k_d - 1$. Both bounding relations can be extended to $C \times C$, defining their value on any other pair of cells by 0. The pACC C is called primal–dual one-dimensional if its primal and dual dimensions both depend on *a primal–dual dimension* function $dms : C \to \{0,1,2,\dots,k\}$, being $k = k_p = k_d$. In fact, $dim_p = dms$ and $dim_d = k - dms$. In this case, let us denote the set of cells C_i^p of primal dimension i simply by C_i and an i-cell mean a primal

i-cell. In this case, the pACC C is called *symmetric* if $B^p(c, c') = B^d(c', c)$, $\forall c, c' \in C$.

Given two cells c and c' of C, we say that the ordered pair (c, c') is *a primal (resp. dual) vector* of the pACC if $B^p(c', c'') \neq 0$ (resp. if $B^d(c', c'') \neq 0$). We say that the set $\{c', c''\}$ is *a primal (resp. dual) interaction* of the pACC if $B^p(c', c'') \neq 0$ or $B^p(c'', c') \neq 0$ (resp. if $B^d(c', c'') \neq 0$ or $B^d(c'', c') \neq 0$). A *link* associated with the primal (resp. dual) vector (c, c') is the set $lnk(c, c')$ of triplets (c, c', c''), for all the cells c'' such that (c', c'') is a dual (resp. primal) vector. A link can also be considered as a sub-pACC of C. $lnk(c, c')$ can be considered as an asymmetric primal–dual one-dimensional pACC, such that its bounding functions $\bar{B}^p$ and $\bar{B}^d$ satisfy an "orthogonality" condition: for all the triplets (c, c', c'') of $lnk(c, c')$, $\bar{B}^p(c, c') = B^p(c, c') \neq 0$, $\bar{B}^p(c', c'') = 0$, $\bar{B}^d(c', c'') = B^d(c', c'') \neq 0$, $\bar{B}^d(c, c') = 0$.

We are able to define the pACC $pACC(I_D) = (Cell(I_D), B^p, B^d, dim_p, dim_d)$ in order to satisfy the following conditions:

- (to be primal–dual one-dimensional) The primal and dual dimensions are both defined in terms of the dimension function *dim*: $Cell(I_D) \to \{0, 1, 2\}$ previously defined in this section.
- (symmetric condition) $pACC(I_D)$ must be symmetric. The primal (resp. dual) bounding relation is defined by $B^p(c', c'') = 1$ (resp. $B^d(c', c'') = 1$), if c' is in the boundary (resp. in the coboundary) of c'' and $B^p(c', c'') = 0$ (resp. $B^d(c', c'') = 0$) otherwise.

Notice that $pACC(I_D) = pACC(I)$ and, therefore, it is a concept dependent of the dimensions of the image I and independent of the ROI D. In an analogous way, we can define another primal–dual symmetric one-dimensional sub-pACC $pACC(D)$ of $pACC(I_D)$, having $Cell(D)$ as set of cells.

In the rest of the paper, *all the pACCs* C *we deal with are primal–dual symmetric one-dimensional pACCs*. A complete theory of combinatorial optimization about pACC-homology and pACC-homotopy operations can be developed in order to provide theoretical robustness to our algorithmic work. This is out of the scope of the present paper and will be discussed by the authors in a future paper.

(c.2) **Installing the set of processing units.** The description of the parallel architecture is the same than that employed in Section 6 of [14]. There are as many processing elements (PE) as pixels the image has (equivalently, as cells $ACC(I_D)$ has). A PE can be specified using the Fig. 2 in which the four mutually 4-adjacent pixels (primal 0-cells of $ACC(I_D)$) are expressed by circles (the processing unit is identified with the pixel colored in orange). The crosses determine the primal 1-cells and the red square is a primal 2-cell. Concerning to the pair of numbers (d, p) associated with each node, d is its dual dimension and p indicates its primal dimension. The topological coordinates of the cells are determined by the symbols $x \in \{1, 2, \ldots, m\}$ and $y \in \{1, 2, \ldots, n\}$. In fact,$(x, y)$ is a 0-cell (pixels of I_D), $(x + \frac{1}{2}, y)$ and $(x, y + \frac{1}{2})$ represent 1-cells and $(x + \frac{1}{2}, y + \frac{1}{2})$ are 2-cells.

Surrounded by a closed curve in yellow, we have four cells $PE_4(x, y) = \{(x, y), (x + \frac{1}{2}, y), (x, y + \frac{1}{2}), (x + \frac{1}{2}, y + \frac{1}{2})\}$ that are the active nodes-cells of the sub-pACC $PE_4(x, y)$ of $ACC(I_D)$ associated with the pixel (x, y). Surrounded by a gray light closed curve, we have all the active nodes and links of the bigger sub-pACC $PE_9(x, y)$ composed by the nine cells. There are two possible states

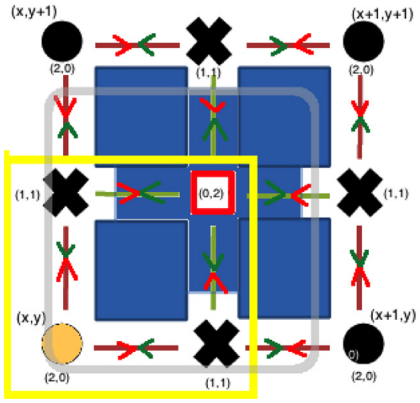

Fig. 2. An specification of a PE. Circles are mutually 4-adjacent pixels; processing unit is identified with the orange pixel. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

for the PE, 0 or 1, depending of the orthogonal privileged direction chosen. The value "0" (resp. "1") means that its updated primal bounding function has value 1 for the primal vectors $\bar{v}_{01} = \{((x, y), (x + \frac{1}{2}, y))\}$ and $\{\bar{v}_{12}\} = \{((x, y + \frac{1}{2}), (x + \frac{1}{2}, y + \frac{1}{2}))\}$ and 0 for the rest (horizontal or East) (resp. by $\bar{w}_{01} = \{((x, y), (x, y + \frac{1}{2}))\}$ and $\{\bar{w}_{12}\} = \{((x + \frac{1}{2}, y), (x + \frac{1}{2}, y + \frac{1}{2}))\}$ and 0 for the rest (vertical or North)). Identifying all the primal vectors $\bar{v}_{01}$, $\bar{w}_{01}$, $\bar{v}_{12}$ and $\bar{w}_{12}$ with its corresponding primal links, we determine in this way all the non-null primal and dual bounding relations in $PE(x, y)$. One rule (called HSF-rule) must be applied in a complete activation of all the PEs of I_D: if initially $B^d(c, c') = 1$, being c a 2-cell of $pACC(I_D)$ and c' a 1-cell of $pACC(I_D)$ involved in a primal 0–1 link of a neighbor (north or east), then $B^d(c, c') = 0$ in the final asymmetric pACC resulting from this global activation of PEs. We assume that the PEs corresponding to the pixels of the north and east border of the image are always activated. They exclusively consist of the link of $\bar{v}_{01}$ (north border) or, correspondingly, the link of $\bar{w}_{01}$ (east border).

If I is represented by a matrix of size $m \times n$, the number of cells of the corresponding primal–dual cell complex $pACC(I_D)$ is $(2n - 1) \times (2n - 1) \approx 4n^2$.

(c.3) **Asymmetric dynamics within huge scenario: Morse Spanning Forests**. In this phase, there are two main PE's activation techniques: (a) Starting from an initial geometric symmetric pACC $pACC(I_D)$, the output of one elementary step of parallel processing is a (non-unique) new particular asymmetric pACC $MrSF(I_D)$, called Morse Spanning Forest (MrSF for short). An MrSF has the property that the set of its elementary primal vectors (or their corresponding links) applied in some order in a sequential process of reduction based on primal homotopy operations provides us a final minimal pACC consisting in one 0-cell. In this way, a MrSF for I_D is seen as a kind of "dense combinatorial skeleton" of the contractible cell complex $Cell(I_D)$. This notion has been already developed in [14] making use exclusively of homological arguments; (b) Starting from a MrSF, the output of one parallel step in this framework is a new MrSF, such that its number of primal links between cells of $Cell(D)$ is greater or equal to that of the initial MrSF. In a later section, a condition for stopping this activation phase is given. The final output is a MrSF presenting a sub-pACC $HSF(D)$ for $Cell(D)$ hav-

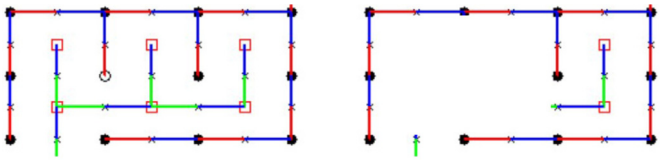

Fig. 3. (Left) Combinatorial MrSF model $MrSF(I_D)$ measuring asymmetric topological dynamics on pACC ambiance: (Right) Homological Spanning Forest $HSF(D)$ of D as sub-pACC of $MrSF(I_D)$.

ing the maximal number of primal links. $HSF(D)$ can also be considered as a Homological Spanning Forest [15].
In Fig. 3, we try to graphically show how to extract a HSF $HSF(D)$ of a ROI from a HSF (in fact, a MrSF) of the ambiance. In general, this simple extraction technique of keeping only pACC information related to cells of the ROI does not work and, as we see in the next section, much more technical work must be done for getting the $HSF(D)$.

3. Morse Spanning Forests

We are able to design a computational method for computing a kind of topological dense skeleton over the pACC $pACC(I_D)$, based on an appropriate reduction of cells via primal homotopy operations.

First, it is necessary to define the notion of primal–dual $(i, i+1)$-path between cells of a (non-necessarily symmetric) pACCs. Given a pACC $\mathsf{C} = (C, B^p, B^d, dim_p, dim_d)$ and an integer $0 \leq i \leq k$, *a primal–dual* $(i, i+1)$*-path* is a sequence $c_1, c_2, \ldots, c_r$ $(r \geq 0)$ of cells of dimensions i and $i+1$ such that any $\{c_j, c_{j+1})\}$ is an elementary interaction of the pACC $(\forall 1 \leq j \leq r-1\})$. Notice that c_1 and c_r can be cells of dimension i or $i+1$. We say that a cell c' is *primal (resp. dual)* $(i, i+1)$*-reachable* from the cell $c \in C_i$ (resp. $c \in C_{i+1}$) if there is a $(i, i+1)$ path starting at c and terminating at c'. This is straightforward to check that primal–dual $(i, i+1)$-reachability is a equivalence relation. Working now with symmetric pACCs, the set of all the equivalence classes with regards primal–dual reachability is called *primal–dual i-pACC–homology set of* C and its cardinality *primal–dual i-pACC-Betti number*. It is straightforward to show that primal or dual pACC-homotopy operations preserve pACC-Betti numbers.

To develop in full detail the theory of pACC-homotopy operations and its power for topological analysis and recognition is beyond the scope of this paper. Here, we limit ourselves to interpret the HSF notion, which has been proved to be useful for topological calculus, [15], in terms of pACC-homotopy operations. With this interpretation at hand, the theory developed in this paper allows us to efficiently compute homology and homotopy invariants (including homotopy groups) of cell versions of digital objects of any dimension and using any adjacency.

A key piece for this translation is a particular kind of asymmetric pACC based on a primal–dual $(i, i+1)$-path $c_1, c_2, \ldots, c_r$, with $c_1 \in C_i$. An analogous dual strategy $(c_1 \in C_{i+1})$ can also be developed. In order to be understandable, we focus only in the primal strategy. The asymmetric pACC naturally constructed from the "sequence" of primal link-pACCs $(c1, c2), lnk(c3, c4), \ldots, lnk(c_{2j-1}, c_{2j})$ $(j = \lfloor r/2 \rfloor)$ allows us to compute maximal (in terms of the number of cells involved) primal pACC-"tree" structures measuring pACC-homology. A pseudo-code for this algorithm is given in Algorithm 1 [3] (Note: $card(S)$ is the cardinal function of a set S).

The output of Algorithm 1 consists in an asymmetric pACC $\mathcal{T}_{(0,1)} + \mathcal{T}_{(1,2)}$, a minimal pACC $\mathcal{H}$ consisting in a set of isolated cells of primal dimension 0 and 1 and a vector field *lnk* of cell pairings. The cells of $\mathcal{H}$ are called *critical cells*. These are the unique cells of C which are not paired by *lnk*. The number of critical cells of dimension 0 and 1 respectively determine the pACC-Betti numbers of C and $B^p_{\mathsf{C}}(c, c') = 1$ for any primal vector (c, c') of *lnk*.

Adding additional constraints to the cell $\bar{c}$ to be chosen in each conditional instruction allow us to get that *lnk* is a primal vector field of *mutually disjoint* pairs of cells. These constraints can be formulated in terms of a function called *oriented flow*. The oriented flow of a cell c of primal dimension i is specified in terms of a particular primal–dual $(i, i+1)$-path "connecting" any cell to a representative cell of its pACC-homology set. In order that the Max-pACC-Forest algorithm provides us of an oriented flow function, we reduce ourselves to say that processes of arrow reversing [6] in *lnk* are compulsory to be done.

In consequence, to describe HSF-structures of cubical cell versions of digital objects or binary digital images mean to apply Max-pACC-Forest algorithm to the pACCs associated with them and to describe later a new redistribution of the primal vectors of the pACC $\mathcal{T}_{(0,1)}, \mathcal{T}_{(1,2)}$, such that two of them in *lnk* are disjoints.

It can be proved that any output of the Max-pACC-Forest algorithm applied to $pACC(D)$ is a HSF of D. For computing an intrinsic feature as $HSF(D)$, we use here a different algorithmic strategy to previous one, which is based on successive modifications of an initial topological model of the ambiance space: a Morse Spanning Forest of digital images.

A *Morse Spanning Forest* (MrSF for short) for a digital image I of dimension $m \times n$ is any output $((\mathcal{T}_{(0,1)}, \mathcal{T}_{(1,2)}), \mathcal{H}, lnk)$ of the Max-pACC-Forest algorithm applied to $pACC(I)$. $\mathcal{T}_{(0,1)}$ and $\mathcal{T}_{(1,2)}$ are respectively called *the 0–1 tree* and *1–2 tree* of the *MrSF*. Given a digital object $D \subset I$ and a MrSF of I_D, a primal 0–1 link $lnk^p(c, c') = \{(c, c', c'')\}$ belonging to *lnk* is called *a sink* (or simply the cell c' is a sink) if the 0-dimensional cell (pixel) c is black (or belongs to D) and the 0-dimensional cell c'' is white (does not belong to D). *A source* of this MrSF is a 1-cell $\{c, c''\}$ of $Cell(I_D)$, such that c and c'' are black pixels and the primal (0, 1)-links $lnk(c, c') = \{(c, c', c'')\}$ and $lnk(c'', c') = \{(c'', c', c)\}$ (being c' the unique 1-cell in $pACC(I_D)$ with $c, c'' \in int^p(c')$) does not belong to vector field *lnk*.

It is straightforward to prove that any complete activation of the PEs corresponding to the parallel architecture of Step (c.2) of Section 2, give raise to a MrSf.

Fig. 5 shows the visualization of two MrSFs $((\mathcal{T}_{(0,1)}, \mathcal{T}_{(1,2)}), \mathcal{H}, lnk)$ associated with a digital image of dimensions 4×4. The existence of a unique critical cell in these particular closed discrete dynamical systems is a consequence of the contractibility of this particular tessellation embedded in the Euclidean plane. The 0–1 (resp. 1–2) primal vectors are colored in red (resp. in blue). The 0–1 (resp 1–2) dual vectors are colored in green (resp. in blue) in the resulting primal 1–2 (resp. 0–1) trees. Both MrSFs show only one 0–1 tree $\mathcal{T}_{(0,1)}$. This directed tree codifies in a smart way the contribution of each primal 0-cell to be part of the unique connected component whose representative element is the critical cell. This contribution is "measured" in terms of a directed path within the tree from the cell to the critical one. There are three 1–2 trees in $\mathcal{T}_{(1,2)}$ on the left and only one in the MrSF on the right.

In order to generate MrSFs presenting only one 1–2 tree in $\mathcal{T}_{(1,2)}$, we must simply apply the algorithm Max-pACC-Forest [3] taking as sub-sequence $c^1_1, \ldots, c^1_{\ell_1}$ of primal 1-cells of C, such that its first 1-cells all belong to the border of the image. In this way, we guarantee that all the border 1-cells of the image excepting one are nodes of the 0–1 tree. From now on, all our MrSFs have this "topological shape".

If D is an object-of-interest within a digital image I of dimensions $m \times n$, a MrSF $((\mathcal{T}_{(0,1)}, \mathcal{T}_{(1,2)}), \mathcal{H}, lnk)$ for I (or, equivalently for I_D) becomes a HSF structure for D if the MrSF superimposed over the image I_D have the maximum number of cells of $Cell(D)$ which are paired by *lnk*. Fig. 4 illustrates some examples of HSF

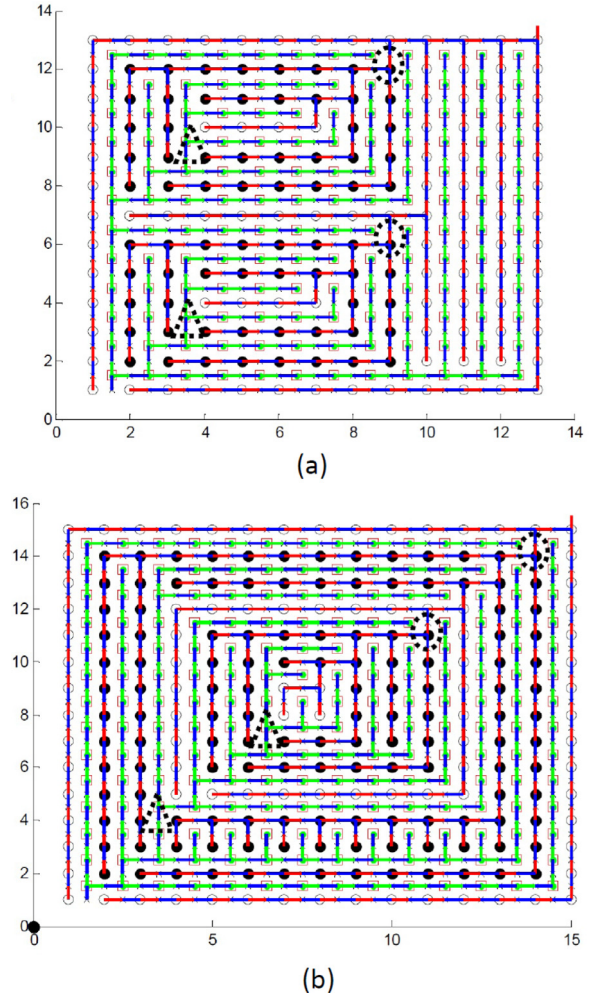

Fig. 4. Examples of HSFs for object of interests with different topologies. (a) An object topologically equivalent to two closed curves; (b) An object topologically equivalent to two concentric closed curves. The sinks are surrounded by dotted ellipses and the sources by dotted triangles.

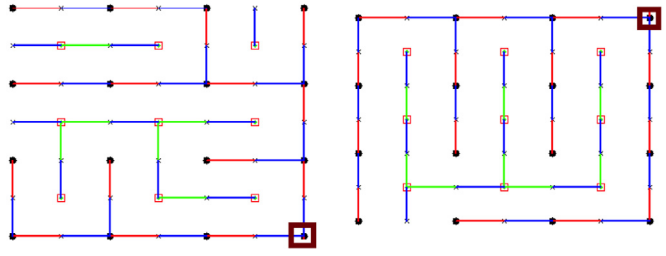

Fig. 5. Two MrSFs for a binary digital image I of dimensions 4×4. Their respective representative cells of the pACC 0-homology of I are surrounded by a hollow square. (For interpretation of the references to colour in the text, the reader is referred to the web version of this article.)

for object of interests with different topologies. The next shapes and their corresponding HSFs are shown: (a) two closed curves, (b) two concentric closed curves. Critical 0-cells (for the ROIs) are marked with a dotted circle and critical 1-cells with a dotted triangle.

Images of Fig. 4 contain objects whose connected components have only one 0–1 sink. This fact guarantee that a true HSF of the foreground (set of black pixels) is obtained. To understand better this problem and to find a possible solution, let us analyze the rotated E shape of Fig. 6. The construction of the MrSF exhibits three

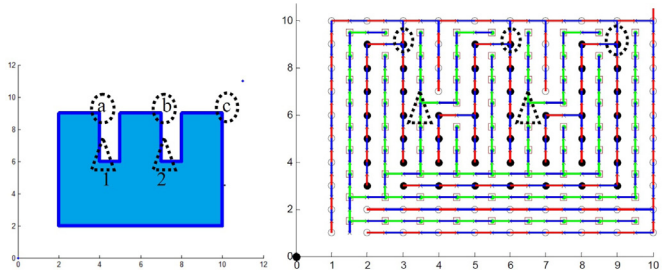

Fig. 6. A rotated E shape showing three sinks with dotted circles and two sources (surrounded with dotted triangles). Two sink/source pairs should be canceled in order to get the HSF.

sinks (tagged a, b, c) and two foreground sources (named 1 and 2, surrounded by dotted triangles). It is obvious that the MrSF of this object must be transformed so as its final HSF would have only one sink.

4. A parallel implementation for obtaining HSF of ROIs

This section includes the sequential and parallel version of the same algorithm: the determination of a HSF of a digital object $D \subset I_D$, being I_D a digital image of dimension $m \times n$. Let us recall that we compute a HSF of D via iterative process of "optimization" of an initial MrSF. Let us assume here that I_D is a binary digital image and that D is the foreground (set of black pixels) of it.

$HSF(D)$ means, in particular, an optimal pairing of cells of D so that it has the minimal possible number of critical (non-paired within $Cell(D)$) 0-cells and 1-cells. Note that in this scenario there will be no critical 2-cells; only critical 0-cells (connected components) and 1-cells (homological holes of dimension 1). All the homological and homotopical magnitudes of the connected components (CC) of the foreground of I_D can be extracted from this HSF. This means that necessarily global information must be extracted, which unavoidably implies to insert some kind of global searching across some points of the whole image. Nevertheless, due to the structure of the MrSF, it is possible to perform large part of the work in parallel. Thus, time complexity is preserved very near to the logarithm of the width plus height of the image. In this sense, we have addressed the codification that it scales well with the number of Processing Elements of the target parallel architecture (like multi or many-core, GPUs, SIMD kernels, FPGA,...). Others software frameworks like REDHOM, [21], are also based on this ideas.

From now on, let us suppose that all the border of whole image is always composed of background pixels. The first step of Algorithm 2 [3] computes the initial MrSF of I_D denoted by $MrSF^0$. The computation of every primal and dual vectors in $MrSF^0$ is exclusively based on the values of its adjacent cells. The rest of steps are necessary to transform this MrSF into a HSF of D (that is, into another MrSF of I_D such that the number of primal links involving cells of D is maximal.

There are different strategies to compute the primal and dual bounding relations for $MrSF^0$. We choose here this strategy based on the activation of PEs (what we call L01): (a) the activation value for the PEs associated with pixels of the west and east (resp. north and south) border of the image is 1 (resp. 0); (b) The activation value for a PE associated with a non-border black (resp. white) pixel (x, y) is 1 if the pixel $(x, y+1)$ is also black (resp. white), it is 0 if the pixel $(x, y+1)$ is white (resp. black) and $(x+1, y)$ is black (resp. white) and it is 1 if the pixels $(x, y+1)$ and $(x+1, y)$ are both white (resp. black). Algorithm 2 shows a sequential implementation for computing $MrSF^0$ based on the previous strategy, whereas Algorithm 3 is its parallel version.

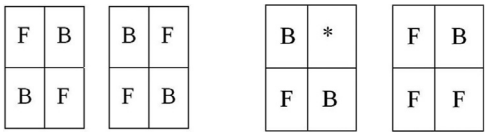

Fig. 7. Left: two crack patterns of 0-cells for 8 adjacent background cells. Middle: Sink pattern of 0-cells. Right: foreground source pattern of 0-cells. F: foreground 0-cells. B: background 0-cells.

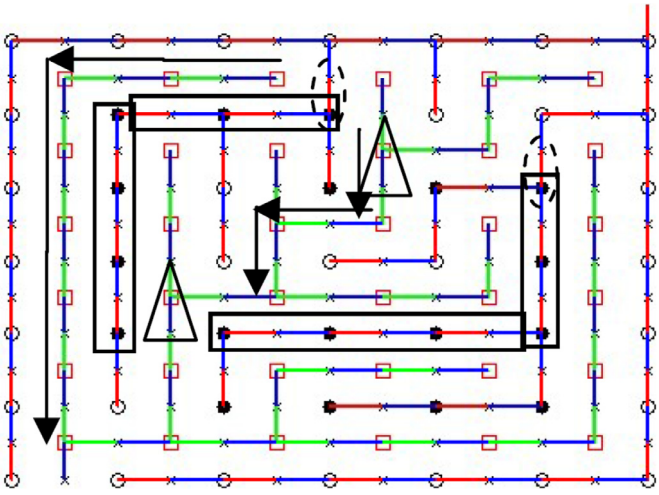

Fig. 8. MrSF for an image with two foreground sinks (marked with dashed ellipses) and two foreground sources (marked with triangles). The paths that some two 0-cells must follow to find their corresponding sinks are drawn with thin rectangles. The paths that two 2-cells must follow to reach their corresponding cracks are drawn with (south or west) arrows. (For interpretation of the references to colour in the text, the reader is referred to the web version of this article.)

This activation of all the PEs can be graphically understood using the example of $MrSF^0$ of Fig. 8. Any red primal vector referred to the vectors $\bar{v}_{01}$ and $\bar{w}_{01}$ is followed in the same direction by blue dual vectors. The 1–2 primal vectors referred to the vectors $\bar{v}_{12}$ and $\bar{w}_{12}$ are colored by blue and 2–1 dual vectors by green. It is straightforward to see that we obtain a MrSF at the end of this process.

The second step of Algorithm 2 determines which 0-cells are background/foreground sinks and which 2-cells are what we have called cracks. We say that a 2-cell is a crack if the 1–2 links that depart from it cross two 4-adjacent foreground 0-cells or two 8-adjacent background 0-cells. This notion of crack is used for fixing the "ends" of some transport path included in the 1–2 tree of $MrSF^0$, allowing afterward the interchanging between a sink and a source. Due to previous criteria for L01, it is straightforward to demonstrate that there are two types of 2-cell cracks for foreground CC: 1) A NS crack happens when a 0-cell has the same color than its adjacent North 0-cell and there is not a primal interaction between them. The crack will be located in the corresponding 2-cell of this 0-cell; 2) A EW crack happens when a 0-cell has the same color than its adjacent East 0-cell and there is not a primal interaction between them. The crack will be located in the corresponding 2-cell of this 0-cell. Besides, for background cells there is another crack case because of the 8 adjacent condition, which correspond to the two patterns of Fig. 7 (left). The crack will be located in the center 2-cell of this pattern. Similarly, sinks are directly identified by the pattern of Fig. 7 (middle), where symbol * means any 0-cell. The 0-cell sink will be located in the foreground 0-cell of this pattern.

Next, the third step of Algorithm 2 is necessary to introduce global relations between 0-cells and the sinks. The aim is determining the sink resulting of following any 0-cell through the vector field *lnk* of $MrSF^0$. Thus, a 0-cell follows its activated primal link $\bar{v}_{01}$ or $\bar{w}_{01}$ and then checks its neighbor point to find out its corresponding activated primal link. This operation is repeated for the next neighbor and so on, until a sink is reached. Finally, each 0-cell can store a label of the sink to which points to. This representation is called here MrSF 0–1 tree. This process is depicted in Fig. 8. The resulting MrSF for this image has two foreground sinks (marked with dashed ellipses). It has been drawn with thin rectangles the paths that two 0-cells must follow to find their corresponding sinks.

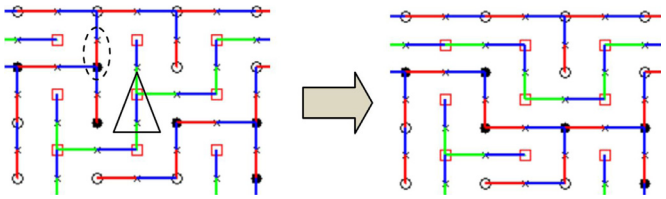

Fig. 9. Left: a fragment of Fig. 8 showing a sink (rounded with a dashed ellipse) with a source (marked with a triangle). Right: the same fragment after the transport that cancels the sink and the source.

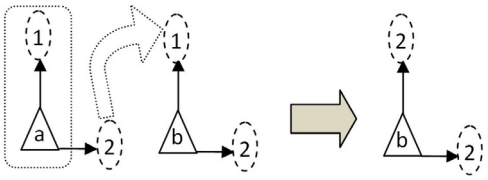

Fig. 10. Left: symbolic notation of east and west sinks pointed by sources of Fig. 8. Right: re-tagging due to the transport process after the cancellation of *a* and 1.

After the 0–1 MrSF tree has been obtained, the MrSF can be converted into a HSF by pairing foreground sources with sinks. It is straightforward to demonstrate that a foreground source exists if and only if the 0-cells form an internal corner in quadrant 1. That is to say, the 2×2 pattern of Fig. 7 right (B means background value and F foreground value). The 1-cell source will be located in between the two foreground bottom 0-cells of this pattern.

A pairing supposes the cancellation of the sink and the source by a path of the 0–1 MrSF tree and the interchanging of its corresponding link patterns. This "transport" can be explained graphically in a straightforward manner (see Fig. 9).

As previously stated, only those 1-cell foreground sources whose east and west 0-cells point to two different sinks in the 0–1 MrSF tree (as that of Fig. 9) are susceptible to be canceled with one of these sinks. On the contrary, if a source pointed to the same sink, it would be the representative 1-cell of a hole, so it must not be canceled. Any transport implies the re-labeling of the 0–1 MrSF tree, which can be explained using the symbolic notation from Fig. 10. In this figure, the sinks and sources of Fig. 8 are named 1, 2 and *a*, *b* respectively, so that the east and west 0-cells represented by *a*, *b* are given by the left graph of Fig. 10. If *a* and 1 are chosen to be canceled, the re-labeling implies that any apparition of the sink 1 in the 0–1 MrSF tree must be changed into 2 (which is the other sink of *a*, as the dashed wide arrow indicates). As a result, the source *b* is now a critical 1-cell that represents the only hole of the black CC in Fig. 8; so it must not be canceled hereafter. This cancellation must be repeated until there are no more can be done. The resulting MrSF (restrained to *D*) provides us the HSF of the image. The remaining sinks and sources are the critical 1-cells and 0-cells of the HSF of the foreground.

Algorithm 3 [3] is the parallel version for determining a HSF of *D* (having the same goal than Algorithm 2). Some steps of the sequential version can be easily parallelized but other steps need a different treatment to promote parallelism.

As we have previously mentioned, the first step of Algorithm 3 is equivalent to the steps 1, 2 of Algorithm 2. Supposing that a processing element is implemented for each pixel, the time complexity order (for a parallel execution) of step 1 would just be O(1).

The second step of Algorithm 3 computes the representation of the 0–1 MrSF tree and also the 1–2 MrSF tree for 2-cells.

The 1–2 MrSF tree is used in the following steps to perform transports in parallel. A reduced number of steps finally give us the final HSF (which obviously contain all the information about the CC contours). An example of the contouring is observed in Fig. 8: the depicted paths (drawn with south or west arrows) of two 2-cells in this figure follow the contour of the foreground CC until they reach a foreground or background crack. Similar paths must be computed for any 0 and 2-cell. It can be seen that these paths follow the contour of each CC in southwest direction until the "end" of a CC has been reached (or a source is found), because the condition for the disappearance of a CC is exactly the existence of a crack. The chosen representations for these trees in this work are the following: every 0-cell stores a label of the first sink that the cell points to, and in the same way every 2-cell stores a label of the first crack that the cell points to. As it can be seen in Algorithm 3(a), the way to proceed starting from a 0-cell (or from a 2-cell) labeling in an efficient parallel manner is to jump with exponentially increasing hops (whose value is in $X12$, $Y12$).

In order to clarify the parallel behavior of the exponentially increasing hops, let us consider a trivial (m, 1)-image having identical cells, and hence with a unique crack in the most south cell. Thus, according to the initialization (step a), for any non-crack cell: $tag12 = 0$, $address = k$ (being $k = 1, 2, \ldots, m-1$ the row index), $Y12 = 1$. On the contrary, the bottom cell will have: $tag12 = m$, $address = m$, $Y12{=}0$. Step b.i gives for each non-crack cell $address = 1 + k$, while the bottom cell stays with $address = m$. After that, step b.ii introduces a first additional nonzero tag in the cell with index $m{-}1$; that is, tag vector will be (0, 0,..., 0, m, m), and it gives a value $Y12 = 2$ for the first $m-2$ cells, by means of $Y12 = Y12 + Y12(address)$. Next iteration will propagate (step b.i) new values for the first $m-2$ cells: $address = k+2$, which means (step b.ii) that tag vector will contain two new nonzero tags, that is: (0, 0,..., 0, m, m, m, m), and the nonzero $Y12$ values will be now equal to 4. In the same way, in the third iteration and for the first $m-4$ cells: $address = k+4$, and Y12 values will be 8. Therefore, tag vector will contain four new nonzero tags, and so on. The increasing jumps propagate the *address* from one cell to its neighboring cells until a sink (crack, correspondingly) is reached. Supposing that a processing element was implemented for each cell, this let us preserve a time complexity order for this step of $O(log(m+n))$.

Using the information of the 0–1 and 1–2 MrSF trees, many transports to the south (step 3) and afterward to the west (step 4) can be done in parallel. This is because the uniqueness for the pairs of sinks/sources along the 1–2 tree of $MrSF^0$ can be guaranteed. Based on Fig. 12, a graphical demonstration of this is given here. In this figure, one sink points to a foreground (1-cell) source along the 1–2 tree of $MrSF^0$ in south direction. With the expression "pointing in south direction" we mean going along the links 1–2 and 2–1 from the nearest southeast 2-cell of the sink and without crossing other crack. The existence of a path from a sink to a foreground source means that there must be a 4-adjacent set of foreground pixels to the "right" of this path (with regards to the flow from sink to source), and an 8-adjacent set of background pixels to its "left". If not, another (background or foreground) source would exist. This supposes a "tube" from the nearest southeast 2-cell of the sink to the source (see dashed lines in Fig. 12). Then, note that the nearest north and east background 0-cells of the considered 0-cell sink are a background crack. Therefore, no other 1-2-1 link path can approach the source from the north without crossing this background crack. To prove that no other sink can reach the foreground source coming from the east, we can proceed by reductio ad absurdum. If there were another foreground sink that points to the same source, another tube must exist with the same flanking. But this new tube is not possible because it must have a set of 4-adjacent foreground pixels to its "right", and this set would cross the background pixels of the other tube.

Likewise, there exist no more than one sink pointing in west direction to a 1-cell foreground source according to a 1–2 MrSF tree of a binary image. The expression "pointing in west direction" means going along the links 1–2 and 2–1 from the nearest northwest 2-cell of the sink and with crossing no other crack. This uniqueness condition can be demonstrated in a similar way to the previous one.

Moreover, note that cancellations done at step 3 (south transports) does not imply any loss of uniqueness for the west transport (step 4). On the contrary, once south and west transports have been done, the 1–2 and 0–1 MrSF trees have changed, so it is not possible to guarantee new similar uniqueness conditions.

Consequently, for each foreground sink, the source of its corresponding southeast 2-cell in the south direction is reached by using the 1–2 MrSF tree. If it is reached a foreground source, transport is executed (see Fig. 9), the sink and the source are "canceled" and the 0–1 and 1–2 MrSF trees must be "re-labeled" (see Fig. 10). The next step is the same than the previous one but we are now looking for the west direction of the west 2-cell of each sink. The uniqueness condition of sink/source pairs guarantees that supposing that a processing element exists for each sink, the time complexity order for the transports in steps 3 and 4 are $O(1)$. However, looking for the corresponding west source for a sink in step 4 needs a searching along the 1–2 MrSF tree, because it was relabeled in the previous step 3. Due to this, a certain number of hops along this tree (until a survival source is reached) must be done in parallel for each sink. Under the assumption that a processing element exists for each 1-cell, the final time complexity order for step 3 and 4 is $O(s)$, where s is the maximum number of hops along the 1–2 MrSF tree in step 4.

Once previous steps have been processed in parallel the 0–1 and 1–2 MrSF trees have been completely re-labeled, thus, it must be checked if the remaining sources are critical 1-cells (representative of holes of the foreground) or simply sources that must be canceled with one of the remaining sinks. Step 5 checks this condition and, at the same time, builds a table with the correspondences between sources and sinks (like the one appearing in Fig. 10, left). In parallel, for the east and west 0-cells of each remaining source, we can access to the re-labeled 0–1 MrSF tree to gather the two corresponding sinks of each source and to build a table or a matrix with these correspondences. Due that previous steps have relabel many elements of 0–1 MrSF tree, the search for the two corresponding sinks of each source supposes a certain number of hops along this tree (until a survival sink is reached). Using this last data structure (a matrix with as many elements as 1-cells), and under the assumption that a processing element exists for each 1-cell, the time complexity order for the step 5 is $O(r)$, where r is the maximum number of hops along the 0–1 MrSF tree.

Finally, step 6 of Algorithm 3 is the same as step 5 in the sequential version, but now it works only for the remaining sinks and sources. For this last part, sequential processing cannot be avoided, but in general it can be shown that the percentage of sink/source pairs involved here is rather small (see Fig. 11). The time complexity order for this final step is $O(q)$, where q is the number of remaining sink/source pairs.

Once the transports and re-labeling of the 0–1 and 1–2 MrSF tree have been performed, the resulting modified MrSF is the desired HSF. The remaining sinks are the critical 0-cells of the foreground, that is, representative elements of the CC. The remaining sources, each of them pointing to the same sink, are the critical

Thumbnail Name at goldminer top40	Width	Height	# initial sources	# hops for west transports	# sources after south and west transports	Max # of hops along the MrSF-0-tree
AJNR2013	1196	1800	1091	235	7	81
PartFibreToxicol-2014	1000	1200	772	24	5	30
AJR2004	295	166	1042	12	101	20
Radiology2010	372	500	987	49	11	126
EURORAD2015	600	498	901	66	1	78
AJR2009	295	260	2364	7	426	17
RadioGraphics2013	482	1274	13	1	0	7
Radiology2007	526	1274	1489	127	7	211
Means	**596**	**872**	**1082**	**65**	**70**	**71**

Fig. 11. Statistical data using the first 8 available medical images of http://goldminer.arrs.org/top-40.php.

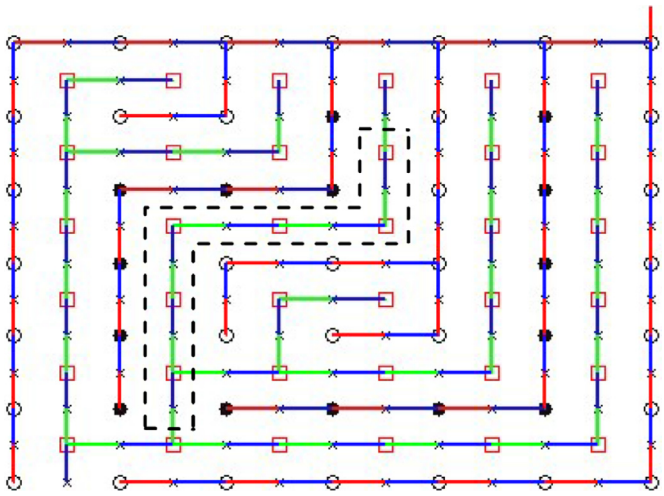

Fig. 12. A "tube" from the nearest southeast 2-cell of the sink to the source (dashed lines), which is flanked by a 4-adjacent set of foreground pixel at its "left", and a 8-adjacent set of background pixel at its "right".

1-cells of D in the HSF, that is, combinatorial representatives of holes.

To sum up, the time complexity order of Algorithm 3 is $O(log(m+n))$ + $O(s)$ + $O(r)$ + $O(q)$. The value of q is different for each image but in the sixth column of Fig. 11 we can see that q means in average approximately a 6.5 % of the initial amount of sources. More clearly, the average number of sequential iterations within step 6 is less than 0.014% of the number of pixels (exactly: 70/(596^*872) = 1,35e−4), or which is more remarkable, less than the 5% (exactly 70/(596 + 872) = 0.048) of $(m+n)$. Likewise, the values of r and s largely varies from one image to another, because it is related to the existence of a "twisted" shape in the CC (like in spirals). The fifth column of Fig. 11 shows that the average value of s is 65, which means also less than the 5% (exactly 65/(596 + 872) = 0.044) of the $(m+n)$. Similarly, the last column shows that the average value of r is 71, which means an inferior value to 5% of the average of $(m+n)$ (71/(596 + 872) = 0.048).

These medical images were binarized by simply selecting a threshold that is the mean of the maximum and minimum gray tones of each image. There is only one image for which the number of iterations is considerably higher than the rest (AJR2009). This is a consequence of the fact that all of its gray tones are very near to the threshold (in fact, its resultant binary image seems to be a pure "salt-and-pepper" image).

In order to declare explicitly the parallelism an implementation [3] in OCTAVE/MATLAB has been first written. All the figures along this work have been generated with these codes. Those

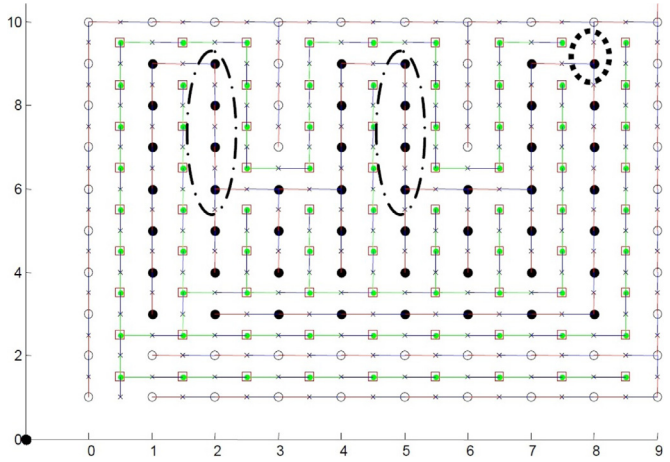

Fig. 13. HSF for a rotated E shape. The discontinued ellipses indicate the paths where earlier MrSF sinks and sources have been canceled and transports have been done. Finally this shape contains only one foreground critical 1-cell (little dotted circle).

matrix operations that cannot be carried out in an element-by-element manner (like matrix inversions, matrix multiply, etc.) have been avoided. Algorithms do not contain any explicit *if...else* conditional sentences. The avoidance of conditional sentences in the "hot spot" zones prevents the so-called thread divergence for GPUs, which is one the main reasons why the performance on this platforms diminishes, [18]. In this sense, some conditional operations have been transformed into predicative-like code, by using firstly element-by-element logical ANDs or multiply operations for the possible results, and second by fusing these results through logical ORs or addition operations.

Hence, this implementation indicates exactly which is the existing data parallelism, and which are the only spots where some loop carried dependencies between iterations exist. As a matter of fact, these spots are only those parts of steps 2, 4, 5, 6 of Algorithm 3, where complexity has been previously revealed as being bigger than O(1), and that are coded as *while* loops. An additional advantage is preventing complex data structures (trees, chained lists, stacks, etc.), which are very inefficient when running on massive data parallel computer architectures (GPUs, SIMD, systolic processors, etc.). Therefore, OpenMP codes can be written directly using this notation, in most of the cases by converting the matrix processing into two nested loops, and preceding the outer loop (devoted to the rows) with the directive *pragma omp parallel for*. Timing results for the OpenMP implementation are shown in next section. Going further, the memory access pattern (which is in most occasions a critical point for the performance that can be achieved in GPUs, [18]) can be clearly observed with the OCTAVE/MATLAB implementation.

Two different examples serve to understand how the preceding algorithms process different shapes. Previous Fig. 6 showed that a rotated E shape contained three MrSF sinks and two foreground sources. These must be canceled (and their corresponding links patterns interchanged) to generate the correct HSF. After applying the previous algorithms, the resultant forest is shown in Fig. 13. The discontinued ellipses indicate the paths where earlier MrSF sinks and sources have been canceled and transports have been done. Finally, this shape contains only one critical 0-cell (little dotted circles), which corresponds to one CC.

Sources and sinks of Fig. 13 are easy to cancel since they lie as consecutive corners. This is the most common case for natural images, where the discretization process usually introduces many stair-like lines that contain many consecutive sinks and sources

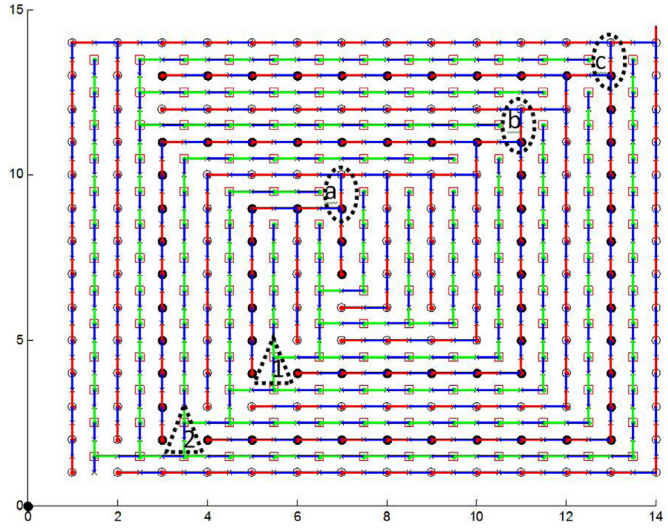

Fig. 14. The MrSF of a spiral shape figure with three MrSF sinks (dotted circles) and two foreground sources (dotted triangles).

Fig. 15. Homotopic relations for Fig. 4(a) (left) and Fig. 4(b) (right).

Thumbnail Name goldminer top40 / #threads	AJNR2013	AJR2004	EURORAD2015	AJR2009
1	0.6573	0.0082	0.0639	0.0151
2	0.3534	0.0035	0.0279	0.0065
3	0.2637	0.0027	0.0176	0.0045
4	0.2290	0.0020	0.0138	0.0037
5	0.2116	0.0016	0.0115	0.0031
6	0.2052	0.0015	0.0098	0.0027
7	0.1920	0.0013	0.0088	0.0023
8	0.1955	0.0011	0.0080	0.0022

Fig. 16. Timing results in seconds for medical images of different size of http://goldminer.arrs.org/top-40.php, using an Intel Xeon E5 2650 server.

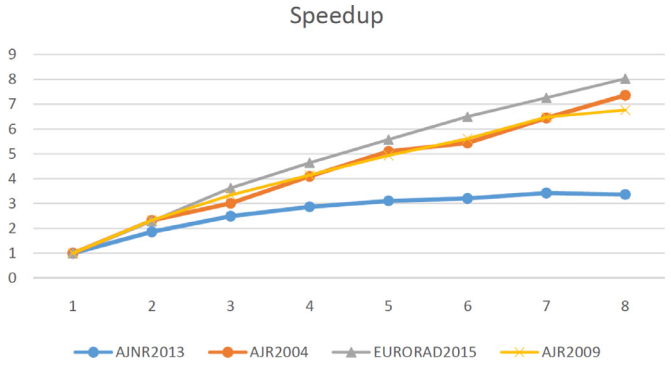

Fig. 17. Speedup for the medical images of previous Fig. 16 vs. the number of threads.

and whose cancellation is straightforward like in the previous case. Nevertheless this is not the case when the curve encloses several windings. Let us consider Fig. 14 the object of interest is a spiral, which actually consists of only one connected component. Fig. 14 shows a draft of the spiral and the MrSF, with three sinks (tagged as *a*, *b*, *c*) and two sources (named 1, 2). For this case, the south and west transports (steps 3 and 4 of Algorithm 3) cannot cancel any sink/source pair because all the primal–dual paths from the sinks fall into background cracks. Thus, the last step 6 of Algorithm 3 must process (in a sequential manner) the last final sink/source pairing, and finally only sink *a* remains as the CC representative 0-cell.

As previously stated, from the HSF of a ROI *D* it is possible to automatically deduce homology and homotopy information of *D* as a cell complex. From the homological point of view the shapes of Fig. 4(a) and (b) are equivalent: their Euler number is 0. From their corresponding HSFs, it is easy to understand that the objects (a) and (b) are different from a homotopy viewpoint. If we label the two critical 0-cell as *a*, *b* and the critical 1-cell as 1, 2, then Fig. 4(a) have the relations of Fig. 15, left, whereas Fig. 4(b) present the relations of Fig. 15, right.

5. Testing results

In order to check the scalability of the algorithms, a non-fully functional but complete implementation has been done in C++ using OpenMP directives [3]. The compiler used was Microsoft Visual Express 12, using flags for full optimization and OpenMP language. Unfortunately this compiler does not produce SIMD-style code (to be executed at the Intel AVX kernels) for the *for* loops.

Initial testing for a PC with an Intel Core i5-4440 (3.10 GHz, 2 × 4 cores, 8 × 32 KB data caches, Level 2 cache size 4 × 256 KB, Level 3 cache size 6 MB, 8 GB RAM) shows that the so-called memory bandwidth bottleneck was reached very early (even for 2 threads) for images above a certain size. This bottleneck appears mainly in the step 2 (the computation of the 0–1 and 1–2 MrSF tree) due the numerous memory accesses that are necessary. This behavior is well predicted by the Roof-line Model [27].

However timing results for another server are very different. This is the case of using an Intel Xeon E5 2650 v2, whose maximum RAM bandwidth is much higher. Its main characteristics are: 2.6 GHz, 8 cores (up to 16 threads), 8 × 32 KB data caches, Level 2 cache size 8 × 256 KB, Level 3 cache size 20 MB, maximum RAM bandwidth: 59.7 GB/s. For this machine, the so-called memory bandwidth bottleneck was reached for images of bigger sizes and for more threads. The mean execution times of 50 repetitions are shown in Fig. 16 for figures of several sizes. Due that the server where tests have been done runs concurrently lots of processes (of different users) distortion appeared in some real time measures. This is more patent for small images and one thread (for bigger times the probability of being interrupted is higher). Due to this, some false superlinear speedup appears for small images. In future testing, our intention is to use a dedicated system that prevents this timing noise.

Scalability (time for various threads divided by time for 1 thread) works pretty well for images where all data structures are kept in L3 caches (see Fig. 17). However, for bigger images scalability decays when the bandwidth RAM bottleneck is reached. However, the almost fully parallelization of our algorithm and these first results seems to point on the direction that the HSF construction may be easily implemented on other parallel architectures. It is expected that the behavior on GPUs could be even better for big images, because GPUs bandwidth to the GDRAM is much bigger than current servers, so the memory bottleneck may probably be reached further.

6. Conclusions

The usefulness and validity of the *Homological Spanning Forest* approach [14,15] within the context of the 2D Digital Imagery is

further consolidated in this paper under the format of primal–dual abstract cell complex. We deliver a parallel framework for topological computation based on HSF structures of images and objects. Both homology and homotopy type information of a ROI D of a digital image I can be efficiently extracted from HSF-models for I. A HSF forest of D is built via parallel and sequential iterative rectification of a HSF of the ambiance space I, increasing in each step the density of cells of the HSF belonging to $Cell(D)$. The technique used in all the elementary steps (excepting the construction of the initial MrSF) of this combinatorial optimization process is based on the **pACC-homology** notion of primal–dual path.

The time complexity order of this framework is close to the logarithm of the sum of the width and the height of the image. Only a linear term (supposing less than its 15% of this sum in the mean) appears on the last steps. Moreover, the software framework is almost fully parallel, so it is expected to scale well for any parallel architecture (GPUs, SIMD kernels, multithreaded, etc.). Finally, in a near future we want to face to the following challenges: the addition of weights to the cells of our abstract primal–dual cell complex definition, because it appears to be an appropriate solution to deal with problems of topological dynamics and physically-based simulation; developing a similar framework for higher dimensional images and ROIs with any kind of adjacency relation.

Acknowledgments

This work has been supported by the Spanish grant (with support from the European Regional Development Fund) BIOSENSE (TEC2012-37868-C04-02/01). The second author acknowledges the support of the FQM-296 Andalusian research group. The last coauthor gratefully acknowledges the support of the Austrian Science Fund (FWF): project number P27516.

References

[1] R. Ayala, E. Domínguez, A.R. Francés, A. Quintero, in: Determining the Components of the Complement of a Digital $(n-1)$-manifold in z^n, Springer-Verlag, London, UK, 1996, pp. 163–176.
[2] F. Chiavetta, V. Di Gesù, Parallel computation of the Euler number via connectivity graph, Pattern Recognit. Lett. 14 (11) (1993) 849–859.
[3] F. Diaz, P. Real, Matlab/octave/c++ codes. http://es.mathworks.com/matlabcentral/fileexchange/56011-topology-hsf-for-images, University of Seville, 2015 (accessed 14.03.16).
[4] E. Dominguez, A.M. Frances, A framework for digital topology, in: Systems, Man and Cybernetics, 1993. 'Systems Engineering in the Service of Humans', Conference Proceedings, vol. 2, 1993, pp. 65–70.
[5] R. Forman, A discrete Morse theory for cell complexes, in: S.T. Yau (Ed.), Topology and Physics for Raoul Bott, International Press, 1995.
[6] R. Forman, Morse theory for cell complexes, Adv. Math. 134 (1998) 90–145.
[7] R. González-Díaz, P. Real, On the cohomology of 3D digital images, Discrete Appl. Math. 147 (2–3) (2005) 245–263. http://dx.doi.org/10.1016/j.dam.2004.09.014.
[8] S. Gupta, D. Palsetia, M.M.A. Patwary, A. Agrawal, A.N. Choudhary, A new parallel algorithm for two-pass connected component labeling, in: 2014 IEEE International Parallel & Distributed Processing Symposium Workshops, Phoenix, AZ, USA, May 19–23, 2014, 2014, pp. 1355–1362.
[9] A. Imiya, U. Eckhardt, The Euler characteristics of discrete objects and discrete quasi-objects, Comput. Vis. Image Understand. 75 (3) (1999) 307–318.
[10] O. Kalentev, A. Rai, S. Kemnitz, R. Schneider, Connected component labeling on a 2D grid using CUDA, J. Parallel Distrib. Comput. 71 (4) (2011) 615–620.
[11] R. Klette, Cell complexes through time, in: Proceedings of SPIE 4117, vol. 4117, 2000, pp. 134–145.
[12] R. Klette, A. Rosenfeld, Digital Geometry—Geometric Methods for Digital Picture Analysis, Morgan Kaufman Series in Computer Graphics, 2004.
[13] R. Lewis, D. Morozov, Parallel computation of persistent homology using the blowup complex, in: Proceedings of the 27th ACM on Symposium on Parallelism in Algorithms and Architectures, SPAA 2015, Portland, OR, USA, June 13–15, 2015, 2015, pp. 323–331.
[14] H. Molina-Abril, P. Real, Homological spanning forest framework for 2D image analysis, Ann. Math. Artif. Intell. 64 (4) (2012) 385–409.
[15] H. Molina-Abril, P. Real, A. Nakamura, R. Klette, Connectivity calculus of fractal polyhedrons, Pattern Recognit. (2014), doi:10.1016/j.patcog.2014.05.016.
[16] J. Munkres, Elements of Algebraic Topology, Addison Wesley, 1984.
[17] A. Murty, V. Natarajan, S. Vadhiyar, Efficient homology computations on multicore and manycore systems, in: 20th Annual International Conference on High Performance Computing, HiPC 2013, Bengaluru (Bangalore), Karnataka, India, December 18–21, 2013, 2013, pp. 333–342.
[18] NVIDIA, Cuda c Best Practices Guide Version, http://developer.nvidia.com/, 2015(accessed 14.07.15).
[19] K. Palágyi, Equivalent 2D sequential and parallel thinning algorithms, in: Combinatorial Image Analysis—16th International Workshop, IWCIA 2014, Brno, Czech Republic, May 28–30, 2014. Proceedings, 2014, pp. 91–100.
[20] P. Pilarczyk, P. Real, Computation of cubical homology, (co)homology and (co)homological operations via chain contractions, Adv. Comput. Math. doi:10.1007/s10444-014-9356-1 (2014).
[21] REDHOM, Redhom, (http://redhom.ii.uj.edu.pl/, 2015, Institute of Computer Science, Jagiellonian University) (accessed 14.07.15).
[22] R. Reina-Molina, D. Díaz-Pernil, P. Real, B. A., Membrane parallelism for discrete Morse theory applied to digital images, Appl. Algebra Eng. Commun. Comput. 26 (1–2) (2015) 49–71.
[23] F. Sergeraert, The computability problem in algebraic topology, Adv. Math. 104 (1994) 1–29.
[24] N. Shivashankar, V. Natarajan, Parallel computation of 3D Morse–Smale complexes, Comput. Graph. Forum 31 (3) (2012) 965–974.
[25] J. Whitehead, Combinatorial homotopy. I, Bull. Am. Math. Soc. 55 (1949) 213–245.
[26] J. Whitehead, Combinatorial homotopy. II, Bull. Am. Math. Soc. 55 (1949) 453–496.
[27] S. Williams, A. Waterman, D.A. Patterson, Roofline: an insightful visual performance model for multicore architectures, Commun. ACM 52 (4) (2009) 65–76.