Parallel $\mathcal{O}(log(n))$ Computation of the Adjacency of Connected Components

Majid Banaeyan(✉) and Walter G. Kropatsch

TU Wien, Pattern Recognition and Image Processing Group 193/03, Vienna, Austria
{majid,krw}@prip.tuwien.ac.at

Abstract. Connected Component Labeling (CCL) is a fundamental task in pattern recognition and image processing algorithms. It groups the pixels into regions, such that adjacent pixels have the same label while pixels belonging to distinct regions have different labels. The common linear-time raster scan CCL techniques have a complexity of $\mathcal{O}(image - size)$ in a 2D binary image. To speed up the procedure of the CCL, the paper proposes a new irregular graph pyramid. To construct this pyramid, we use a new formalism [1] that introduces an order of the pixels in the base grid to detect the redundant edges through the hierarchical structure. These redundant edges, unlike the usual methods of constructing the irregular pyramid, are removed before contracting the edges. This not only simplifies the construction processes but may decrease memory consumption by approximately half. To perform the CCL task efficiently the proposed parallel algorithm reduces the complexity to $\mathcal{O}(log(n))$ where the n is the diameter of the largest connected component in the image. In addition, using an efficient combinatorial structure the topological properties of the connected components including adjacency of CCs, multi-boundaries and inclusions are preserved. Finally, the mathematical proofs provide fully parallel implementations and lead to efficient results in comparison with the state-of-the-art.

Keywords: Connected Component Labeling · Irregular graph pyramid · Parallel processing · Combinatorial map · Pattern recognition

1 Introduction

Connected Component Labeling (CCL) is used in analysing binary images as a basic task [16]. Given as input a binary image, its values distinguish between background (zero) or foreground (one) regions. After this, a region is **connected** if all pairs of pixels are connected by a chain of neighbors. They may be multiple regions with value zero and multiple regions with value one. CCL assigns a unique label to each different region. In general, the CCL algorithms divide into

Supported by Pattern Recognition Image Processing (PRIP) group, Vienna, Austria.

M. El Yacoubi et al. (Eds.): ICPRAI 2022, LNCS 13364, pp. 102–113, 2022.
https://doi.org/10.1007/978-3-031-09282-4_9

two main categories [11] based on label-propagation [10] or label-equivalence-resolving [12]. All of these approaches are linear and a pixel usually is visited in the raster-scan search. In other words, such algorithms may differ from one-scan or two-scan searching through the entire image, but all of them are in the order of image size, $\mathcal{O}(MN)$ in a $M \times N$-sized (2D) binary image. Recently, the algorithm proposed in [3] uses a pyramid structure for the CCL. However, because of the linear propagation of the labels, it is linear as well.

In contrast, In this study, the proposed Parallel Pyramidal Connected Component (//ACC[1]) method reduces the complexity impressively to the logarithmic order of the diameter of the largest connected component in the image. To this aim, we employ a new formalism in [1] to recognize the redundant edges in the pyramid. Removing these redundant edges *before* contracting the edges, not only is performed in parallel but may decrease memory consumption by half in comparison with efficient pyramids [17].

To construct the irregular pyramid, the **R**emove **t**hen **C**ontract (RtC) algorithm is proposed. The proposed algorithm speeds up the labeling task which makes it more efficient to be used in various application areas of machine learning and artificial intelligence such as document analysis and object recognition [15].

1.1 Motivations and Notations

Irregular pyramids are a stack of successively reduced graphs where each graph is constructed from the graph below by selecting a specific subset of vertices and edges. For generation of irregular pyramids, two basic operations on graphs are needed: edge contraction and edge removal. The former contracts an edge $e = (v, w)$, identifies v and w and removes the edge. All edges that were incident to the joined vertices will be incident to the resulting vertex after the operation. The latter removes an edge from the graph, without changing the number of vertices or affecting the incidence relationships of other edges. Note that in this study for preserving topology the self-loops are not contractable. In each level of the pyramid, the vertices and edges disappearing in level above are called *non-surviving* and those appearing in the upper level *surviving* ones.

Definition 1 (Contraction Kernel (CK). *A contraction kernel is a spanning tree of the connected component with the surviving vertex as its root.*

Each contraction kernel is a tree including one surviving vertex and the remaining non-surviving vertices of the CC.

A ***plane*** graph is a graph embedded in the plane such that its edges intersect only at their endpoints [18]. In plane graph there are connected spaces between edges and vertices and every such connected area of the plane is called a *face*. The *degree* of the face is the number of edges bounding the face. In addition a face bounded by a cycle is called an *empty face*. In a non-empty face traversing the boundary would require to visit vertices or edges twice.

[1] It is pronounced pac where the // and A stand for parallel and pyramidal.

There are different structures to build the irregular pyramid such as simple graphs [7], dual multi-graphs [13] and combinatorial maps (CM) [6]. The simple graph cannot distinguish some topological configurations (inclusions and multiple adjacency) [14]. The problem with dual graphs is that they cannot unambiguously represent a region enclosed in another one on a local level [7]. Therefore, in this paper the CM is used which not only resolves the mentioned problems but also can be extended to higher dimensions (nD).

A **combinatorial pyramid** is a hierarchy of successively reduced combinatorial maps [6]. A combinatorial map (CM) is similar to a graph but explicitly stores the orientation of edges around each vertex. To this aim, a permutation, σ, is defined encoding consecutive edges around a same vertex while turning counterclockwise. The clockwise orientation is denoted by σ^{-1}. In the CM each edge divides into two half-edges. Each half-edge is called a *dart* and the α is an *involution* providing a one-to-one mapping between consecutive darts forming the same edge such that $\alpha(\alpha(d)) = d$.

Figure 1 left, shows a set of 8 adjacent darts with their σ relations in a face of degree 4. In the middle, it shows the encoding of the darts. For instance, consider $e = (1, 2)$ where $\alpha(1) = 2$, $\alpha(2) = 1$, $\sigma(1) = 5$. In this paper, we assign an odd number to the left-side dart of a horizontal edge while assigning an even number to its right-side dart. Similarly, we assign an odd number to the up-side dart of a vertical edge while assigning an even number to its down-side dart. The d_{odd} indicates an odd dart and the d_{even} indicates an even dart (Fig. 1 right).

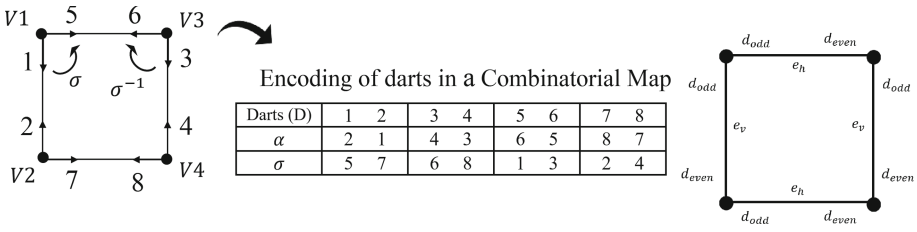


Fig. 1. Combinatorial map [1]

2 The RtC Algorithm for Pyramid Construction

In the original irregular pyramids [7] selected edges are first contracted. Edge contraction has the main advantage to preserve the connectivity. But it has a side effect to produce multiple edges and self-loops. Some of these edges are necessary to properly describe topological relations like inclusions and multiple connections between the same vertices. However, many of them are not necessary and hence called *redundant*. Redundant edges are removed through the simplification procedure after the contractions. However, in the proposed **R**emove **t**hen **C**ontract **(RtC)** algorithm, the redundant edges are removed *before* the contractions and in a parallel way. To this aim, the RtC introduces a new formalism to

define the redundant edges (Sect. 2.3). Since the RtC is used for the CCL task, the input is a binary image where the 4-connectivity between pixels is assumed because the 8-connectivity would not create a plane graph.

2.1 Edge Classification

Consider the neighborhood graph $G(V, E)$ of a binary image P where the vertices V correspond to the pixels P and the edges E connect two vertices if the corresponding pixels are 4-neighbors. Let the gray-value of vertex $g(v) = g(p)$ where $p \in P$ is a pixel in image corresponding to vertex v. Let $contrast(e)$ be an attribute of an edge $e(u, v)$ where $u, v \in V$ and $contrast(e) = |g(u) - g(v)|$. Since we are working with binary images, the pixels (and corresponding vertices can) have only two gray values 0 and 1. Similarly the edge contrast can have only two possible values 0 and 1. The edges in the neighborhood graph can be classified into the following two categories:

Definition 2 (Zero-edge) *An edge is a zero-edge iff the contrast between its two endpoints is zero. The zero-edge is denoted by* e_0.

Definition 3 (One-edge) *An edge is a one-edge iff the contrast between its two endpoints is one. The one-edge is denoted by* e_1.

The set of edges classified as zero-edge is denoted as E_0 and the set of edges classified as one-edge is denoted as E_1. The edge set $E = E_0 \cup E_1$.

A connected component consists of E_0 and E_1 connects different CCs together. Thus, the proposed algorithm for doing the labeling task, only considers E_0 as the candidates of the selection of the CK. Figure 2 shows a binary image with its corresponding neighborhood graph. Edges E_0 are black while E_1 are red.

2.2 Selecting the Contraction Kernel

The way a CK is selected has a main role in detecting the redundant edges in the neighborhood graph. To this purpose, a total order defined over the indices of vertices. Consider the binary image has M rows and N columns such that $(1, 1)$ is the coordinate of the pixel at the upper-left corner and (M, N) at the lower-right corner. An index $Idx(.,.)$ of each vertex is defined:

$$Idx : [1, M] \times [1, N] \mapsto [1, M \cdot N] \subset \mathbb{N} \tag{1}$$

$$Idx(r, c) = (c - 1) \cdot M + r \tag{2}$$

where r and c are the row and column of the pixel(v), respectively. Since the set of integers is totally ordered each vertex has a unique index. The important property of such totally ordered set is that every subset has exactly one minimum and one maximum member (integer number). This property provides a unique orientation between non-surviving and surviving vertices. Consider a non-surviving vertex v. In order to find the surviving vertex, v_s, an incident

e_0 must be found in its neighborhood. Such a neighborhood $\mathcal{N}(v)$ is defined as follows [1]:

$$\mathcal{N}(v) = \{v\} \cup \{w \in V | e_0 = (v, w) \in E_0\} \tag{3}$$

if such neighborhood exists ($|\mathcal{N}(v)| > 1$) the surviving vertex is:

$$v_s = \text{argmax}\{Idx(v_s) |\ v_s \in \mathcal{N}(v), |\mathcal{N}(v)| > 1\} \tag{4}$$

Definition 4 (Orientation of a e_0). *A $e_0 = (v, w) \in E_0$ is oriented from v to w if w has the largest index among the neighbors, $Idx(w) = \max\{Idx(u) | u \in \mathcal{N}(v)\}$. All edges to the other neighbors remain non-oriented.*

Based on the definition above, a chain of oriented edges connects each non-surviving vertex to its corresponding survivor vertex. In Fig. 2 the oriented edges are represented by an arrow over each e_0. The surviving vertices (4, 12, 15), are presented by a green circle around each one.

In this paper, vertices surrounded by only $e_1 \in E_1$ are ***isolated vertices***. The isolated vertices will not be contracted through the construction process and they survive until the top of the pyramid. In the Fig. 2 the isolated vertices are 10 and 16 indicated by a blue circle.

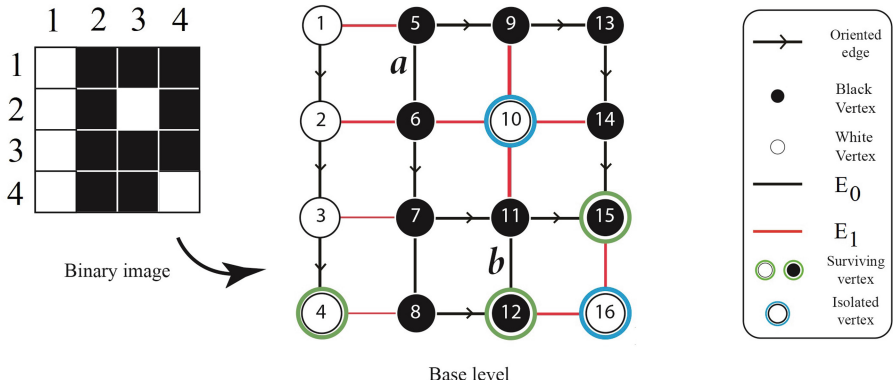


Fig. 2. The neighborhood graph of a 4 by 4 binary image.

2.3 Redundant Edges

In [1], redundant edges are investigated in details. To construct the irregular pyramid based on the RtC algorithm, first, the redundant edges are defined.

Definition 5 (Redundant-Edge (RE)). *In an empty face, the non-oriented edge incident to the vertex with lowest Idx is redundant iff:*

- *The empty face is bounded by only non-oriented edges with the same contrast value.*

– *The empty face is bounded by non-oriented edges with the same contrast value and oriented edges.*

Proposition 1. *The upper bound of the maximum number of redundant edges (REs) is equal to half of the edges of the grid at base level.*

Proof. Can be found in [1]. □

Since edges classify into E_0 and E_1, the Redundant Edges (REs) are partitioned into *Redundant Zero-Edges* (RE_0) and *Redundant One-Edges* (RE_1) as well:

$$RE = RE_0 \dot{\cup} RE_1 \tag{5}$$

Removing RE and contracting the selected CKs at the base level, result in building the first level of the pyramid. To build the upper levels, the CKs are selected and then are contracted until there is no edge remaining for contraction. At this point, the pyramid reaches to its top level and the RtC algorithm is terminated.

In Fig. 3, different levels of the pyramid are shown. At the base level, the RE_0 is shown by a black dashed-line and the RE_1 are shown by red dashed-lines. Furthermore, the Region Adjacency Graph (RAG) of the middle and top level are illustrated. The RAG at top of the pyramid represents the connections between four different connected components. Using the combinatorial map structure, the **inclusion** relation is preserved as it is represented by the loop **a** around the vertex 10. Additionally, the structure preserves the **multiple boundaries** as it is shown by two different edges between vertices 4 and 15 with different paths of vertices (4-3-2-1-5-9-13-14-15 and 4-8-12-11-15) from the base level.

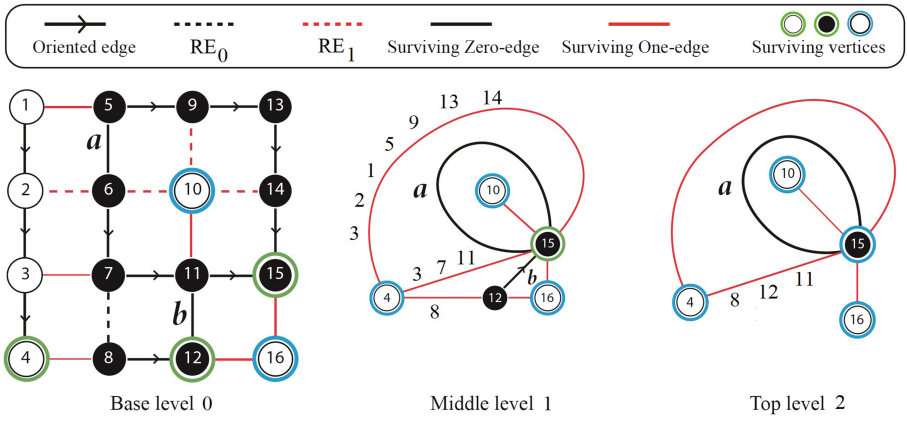


Fig. 3. Binary irregular pyramid. (Color figure online)

2.4 Parallel Pyramidal Connected Component (//ACC)

The goal of connected component labeling is to assign a unique label to the vertices of a CC at the base level. Given a binary image as an input, first the corresponding pyramid is built by the RtC. At the top of the constructed pyramid, the RAG presents connected components (CCs) and the connectivity relations. Each CC is represented by one surviving vertex. The range of vertices between 1 to $M.N$ is kept. For each vertex a label as a new attribute is initialized. A surviving vertex at the top uses its index Idx as its unique label. To propagate down, each non-surviving vertex below the top level checks its parent and fills the label with the label of the parent. By reaching to the base level all the vertices receive their labels and the labeling task is finished. Since the CCL task is performed using the pyramid structure and in parallel, we call it **P**arallel **P**yramidal **C**onnected **C**omponent (**//ACC**).

3 Parallel Complexity

In this section the parallel complexity of the proposed //ACC algorithm is investigated. Whenever we talk about complexity, it is always assumed parallel complexity. The size of the binary input image is $M \times N$. Therefore, the indices of the vertices and the neighborhood relations of the edges are known. Note that such indexing is available *before* constructing the pyramid and in off-line processes. The edge classification and selection of the CKs are both performed locally over a vertex and its neighborhoods and therefore in parallel.

To remove the redundant edges (RE), a dependency between edges is considered. We define such dependency relation to detect a set of redundant edges where by simultaneously removing, the combinatorial structure is not harmed. Therefore, first a set of dependent darts is defined as follows:

Definition 6 (Dependent Darts). *All darts of a σ-orbit sharing an endpoint are dependent darts.*

Afterwards, by using the corresponding edge of each dart, $e = (d, \alpha(d))$, the set of dependent darts leads to the set of **dependent edges**. As a consequence, two edges not sharing an endpoint are independent. In this way, the only case of the dependency between RE occurs when the RE share an endpoint.

In the grid at the base level the RE may be horizontally or vertically connected and therefore are not independent. However, consider a horizontal edge in an odd row of the grid. This edge is independent to all other horizontal edges of other odd rows. Similarly, a vertical edge in an odd column is independent to all other vertical edges of other odd columns. Such independency occurs between edges in even rows and even columns as well. Figure 4.a , represents the set of independent edges at the base. Thus, all the edges in grid are classified into four independent set of edges.

As a result, removing all edge belong to each independent set (1, 2, 3 or 4), occurs simultaneously. This means, all the RE are removed in only four steps

where each step has the complexity $\mathcal{O}(1)$. Therefore removing the redundant edges is performed in parallel.

The disjoint sub-trees of a CK do not share any edges and therefore their contractions are performed in parallel. The challenging task is how to contract edges inside a tree of a CK in parallel? To this aim, we introduce two methods, one only for the base and the other for the remaining levels of the pyramid.

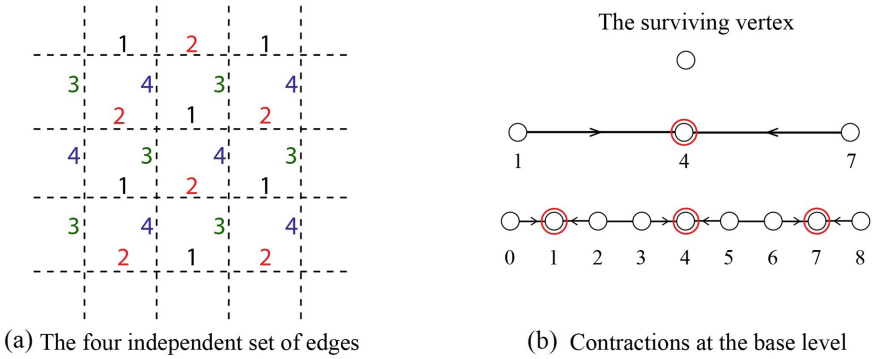
Contractions at the Base Level: Note that the diameter of a CK is the length of the largest path in the CK.

Proposition 2. *The complexity of contracting a CK has a logarithmic bound as follow:*

$$log_2(\delta(CK)) \leq complexity \quad of \quad contracting \quad a \quad CK \leq log_3(\delta(CK)) \quad (6)$$

where the $\delta(CK)$ is the diameter of the CK of the largest connected component in the image.

Proof. Based on (4), the maximum diameter of an oriented sub-tree graph corresponding to a $M \times N$ image is equal to $M + N - 1$. We consider a line sequence of edges with its length equal to this diameter. Next, the numbers from 0 to $M + N - 2$ are assigned to the vertices of the line sequence. By choosing the survivor vertices at $3n+1$ ($n \in \{0, 1, 2, [(M+N-2)/3]\}$), adjacent non-survivors ($3n$ and $3n + 2$) are contracted to this survivor (Fig. 4.b). Since each survivor belongs to a CK with the diameter at most 2, a line sequence consists of $3k$ vertices where $k \in \{1, 2, 3, ...\}$, needs only $log_3(3k)$ steps to select the survivors. The worst case occurs when the length of the line sequence is 4 and therefore, two steps ($log_2(4)$) are required for selecting the survivors. □



(a) The four independent set of edges (b) Contractions at the base level

Fig. 4. Independent edge sets [1], and contractions at the base

Contractions at Upper Levels: Based on (4), all remaining non-oriented E_0 are vertical edges at the base level (Fig. 5.b). The E_0 of two different CCs are

disjointed and therefore they are independent. The E_0 of a CC may have at the worst case the $N-1$ vertical non-oriented edges in the neighborhood graph of the M by N binary image (CC1 in Fig. 5.b). These non-oriented edges receives their orientations at the next level ($L=1$) through the procedure of selecting the CKs and create a line of oriented edges. Such the line sequence of oriented E_0 are contracted in $\mathcal{O}(log_2(N-1))$ as visually is encoded in Fig. 5.c.

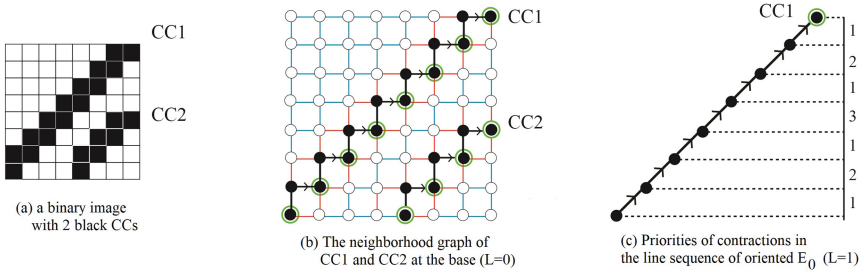


Fig. 5. Priorities of contractions at level 1

4 Comparisons and Results

Simulations use MATLAB software and execute over CPU with AMD Ryzen 7 2700X, 3.7 GHz. The YACCLAB [9] benchmark was used for evaluating the proposed algorithm. The algorithm is executed over 89 random, 128 MRI and 128 finger-print images from this benchmark. Table 1 shows the results.

Table 1. Results over images of different categories from (YACCLAB[9]).

Database type	Random images	MRI images	Finger-print images
Size of the image	128 × 128	256 × 256	300 × 300
Redundant edges (average)	27.66%	46.49%	46.05%
Redundant edges (worst case)	23.18%	44.42%	42.50%
Number of connected components	2192	691	543
Execution time (ms) (in average)	0.098	1.643	2.317
Execution time (ms) (worst case)	0.127	2.973	3.518

The average percentage of the redundant edges (RE) over each category is represented. For example in the finger-print images, about 45% of the edges are redundant while they are all removed in parallel. Moreover, the number of connected components and the average time in each category are shown. The category of Random Images consist of only small objects. It means the diameter of a CK of the largest connected component of these small objects is negligible

in compare to the diameter of the image. Therefore the complexity is near to the $O(1)$. Essentially, the worst case occurs when the size of an object is as large as the whole image. In such the case, the complexity is equal to the logarithmic of the diameter of the image.

The inclusion relationship (hole) is one of the important topological information between connected components. The implementation of the proposed labeling //ACC, not only performs the labeling task, but also provides the number of inclusions between connected components. Furthermore, the simulations represent the adjacency and multi-adjacency of CCs. Such valuable topological information are missing in usual CCL algorithms. Figure 6 shows the CCL over a binary mitochondria image. The corresponding graph of the base level and categories of the edges are illustrated. The image consists of 9 connected component where the inclusion number is 7. In addition, the number of different edges for the mitochondria image are compared. The experimental results show approximately half of the edges in this image are RE.

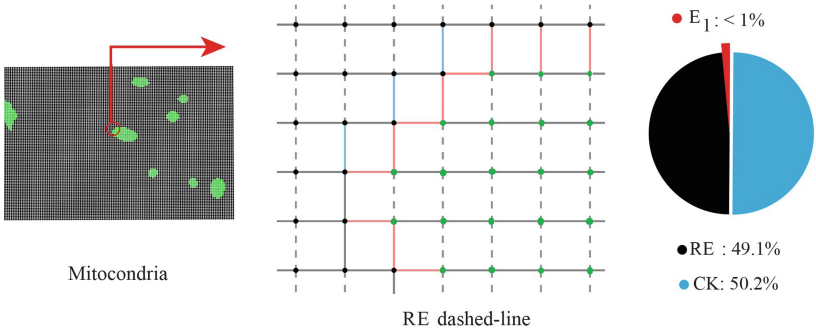


Fig. 6. A binary mitochondria image from [9]. Number of CCs is 9. The number of inclusions (holes) is 7. The RE are almost half of the edges.

Figure 7 shows the execution time of the //ACC algorithm over different image-sizes and compares it with the state-of-the-art methods from [5]. Although for small images the efficient algorithms in [5] are executed in higher speeds the //ACC with its logarithmic complexity reaches to the faster labeling results for big data, i.e., images larger than one million pixels.

Removing the RE not only speeds up the execution, but also decreases the memory consumption. The comparison is done with the originally proposed canonical represented [17] that also is used in [2,4,8]. In canonical representation, the minimum storage required to store the structure is equal to the number of darts i.e. twice the number of edges. By using the proposed RtC method, we eliminate the edges that are structurally redundant and consequently reduce the storage space of darts. Since the upper bound of the maximum number of RE is equal to half of the edges, the memory consumption of the proposed algorithm may decrease approximately by half.

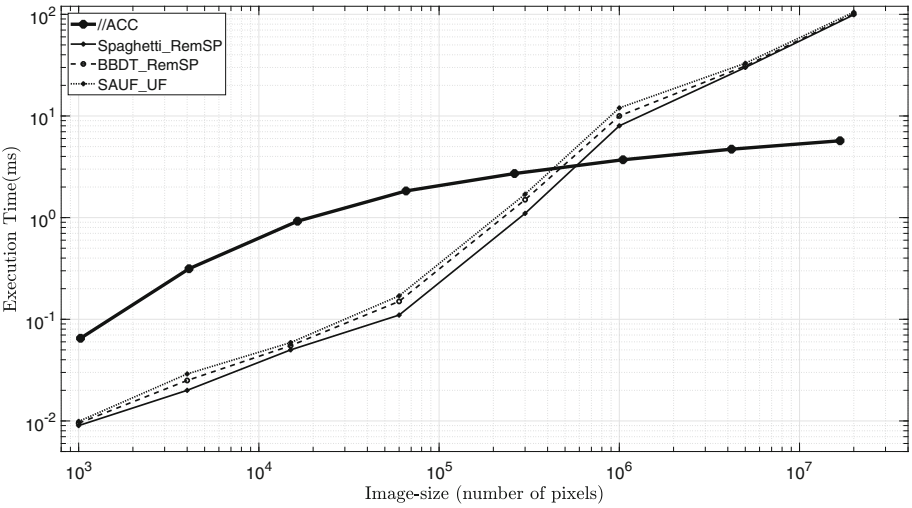


Fig. 7. Illustration of the execution time (ms) over different image-sizes

5 Conclusions

The paper presented a new approach to construct the irregular graph pyramids such that the connected component labeling can be performed in parallel and therefore faster. Unlike the usual construction of the irregular pyramids, in this paper, the redundant edges were removed in parallel *before* the contractions while they used to be removed after contractions and in a sequential order. The experimental results show that nearly half of the edges are removed as redundant edges that decreases the memory consumption to half of the combinatorial map of the base level of the pyramid. The logarithmic complexity of the algorithm speeds up the execution and suits it particularly for large images. In addition, the proposed method provides additional topological information such as inclusion and multi-boundaries. Moreover, what we proved it seems to be true for general graphs. Finally, using the combinatorial structure the proposed connected component labeling method can be extended to higher dimensions (nD) and to multi-label segmented images.

References

1. Banaeyan, M., Batavia, D., Kropatsch, W.G.: Removing redundancies in binary images. In: International Conference on Intelligent Systems and Patterns Recognition (ISPR), Hammamet, Tunisia, 24–25 March. Springer, Heidelberg (2022). (in print)
2. Banaeyan, M., Huber, H., Kropatsch, W.G., Barth, R.: A novel concept for smart camera image stitching. In: Čehovin, L., Mandeljc, R., Štruc, V. (eds.) Proceedings of the 21st Computer Vision Winter Workshop 2016, pp. 1–9 (2016)

3. Banaeyan, M., Kropatsch, W.G.: Pyramidal connected component labeling by irregular graph pyramid. In: 2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA), pp. 1–5 (2021)
4. Batavia, D., Gonzalez-Diaz, R., Kropatsch, W.G.: Image = Structure + Few colors. In: Torsello, A., Rossi, L., Pelillo, M., Biggio, B., Robles-Kelly, A. (eds.) S+SSPR 2021. LNCS, vol. 12644, pp. 365–375. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-73973-7_35
5. Bolelli, F., Allegretti, S., Baraldi, L., Grana, C.: Spaghetti labeling: directed acyclic graphs for block-based connected components labeling. IEEE Trans. Image Process. **29**, 1999–2012 (2020)
6. Brun, L., Kropatsch, W.: Combinatorial pyramids. In: Proceedings of International Conference on Image Processing, vol. 2, pp. II-33. IEEE (2003)
7. Brun, L., Kropatsch, W.G.: Hierarchical graph encodings. In: Lézoray, O., Grady, L. (eds.) Image Processing and Analysis with Graphs: Theory and Practice, pp. 305–349. CRC Press (2012)
8. Cerman, M., Janusch, I., Gonzalez-Diaz, R., Kropatsch, W.G.: Topology-based image segmentation using LBP pyramids. Mach. Vis. Appl. **27**(8), 1161–1174 (2016). https://doi.org/10.1007/s00138-016-0795-1
9. Grana, C., Bolelli, F., Baraldi, L., Vezzani, R.: YACCLAB - yet another connected components labeling benchmark. In: 2016 23rd International Conference on Pattern Recognition (ICPR), pp. 3109–3114. Springer, Heidelberg (2016)
10. He, L., Chao, Y., Suzuki, K.: Two efficient label-equivalence-based connected-component labeling algorithms for 3D binary images. IEEE Trans. Image Process. **20**(8), 2122–2134 (2011)
11. He, L., Ren, X., Gao, Q., Zhao, X., Yao, B., Chao, Y.: The connected-component labeling problem: a review of state-of-the-art algorithms. Pattern Recogn. **70**, 25–43 (2017)
12. Hernandez-Belmonte, U.H., Ayala-Ramirez, V., Sanchez-Yanez, R.E.: A comparative review of two-pass connected component labeling algorithms. In: Batyrshin, I., Sidorov, G. (eds.) MICAI 2011. LNCS (LNAI), vol. 7095, pp. 452–462. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25330-0_40
13. Kropatsch, W.G.: Building irregular pyramids by dual graph contraction. IEE-Proc. Vis. Image Signal Process. **142**(No. 6), 366–374 (1995)
14. Kropatsch, W.G., Macho, H.: Finding the structure of connected components using dual irregular pyramids. In: Cinquième Colloque DGCI, pp. 147–158. LLAIC1, Université d'Auvergne, ISBN 2-87663-040-0 (1995)
15. Qin, H., El Yacoubi, M.A.: End-to-end generative adversarial network for palm-vein recognition. In: Lu, Y., Vincent, N., Yuen, P.C., Zheng, W.-S., Cheriet, F., Suen, C.Y. (eds.) ICPRAI 2020. LNCS, vol. 12068, pp. 714–724. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59830-3_62
16. Shapiro, L.G.: Connected component labeling and adjacency graph construction. Mach. Intell. Pattern Recogn. **19**, 1–30 (1996)
17. Torres, F., Kropatsch, W.G.: Canonical encoding of the combinatorial pyramid. In: Proceedings of the 19th Computer Vision Winter Workshop, pp. 118–125 (2014)
18. Trudeau, R.: Introduction to Graph Theory. Dover Books on Mathematics (1993)