1-Attempt 4-Cycle Parallel Thinning Algorithms

Kálmán Palágyi[a] and Gábor Németh[b]
Department of Image Processing and Computer Graphics, University of Szeged, Szeged, Hungary

Keywords: Shape Representation, Skeletonization, Thinning, Digital Topology.

Abstract: Thinning is a frequently applied skeletonization technique. It is an iterative object-reduction in a topology-preserving way: the outmost layer of an object is deleted, and the entire process is repeated until stability is reached. In the case of an 1-attempt thinning algorithm, if a border pixel is not deleted in the very first time, it cannot be deleted in the remaining phases of the thinning process. This paper shows that two 4-cycle parallel 2D thinning algorithms (i.e., one subiteration-based and one subfield-based) are 1-attempt. In addition, we illustrate that both algorithms are considerably faster if we know that they fulfill the 1-attempt property.

1 INTRODUCTION

A *binary digital picture* (*picture* in short) on the 2D square grid is composed of *black* or *white* unit squares that are called as *pixels*. A *parallel reduction* transforms a picture only by changing some set of black pixels to white ones at a time, which is referred to as *deletion* (Hall, 1996).

Skeletonization in 2D provides *centerline* that is a compact representation of a binary object, and it is used in various image processing and computer vision applications (Saha et al., 2017; Siddiqi and Pizer, 2008).

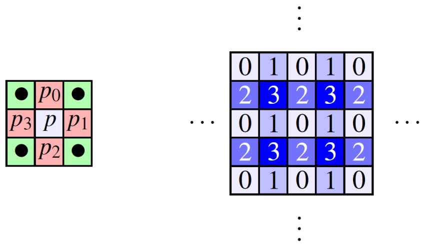
Thinning is an iterative skeletonization technique: border pixels of a binary object that satisfy certain topological and geometric constraints are deleted, and this process is repeated until no pixels are deleted (i.e., the centerline is reached) (Hall, 1996; Kong and Rosenfeld, 1989). Parallel thinning algorithms are composed of parallel reductions, and they fall into three major categories: fully parallel, subiteration-based (or directional), and subfield-based (Hall, 1996; Németh and Palágyi, 2011). *Fully parallel* algorithms apply the same parallel reduction in each iteration; in *subiteration-based* algorithms a cycle of $d \geq 2$ parallel reductions are assigned to the selected d kinds of deletion directions, and only border pixels of a certain kind can be deleted at a subiteration; *subfield-based* algorithms partition $\mathbb{Z}^2$ into $s \geq 2$ subsets which are alternatively activated, and only some pixels in the active subfield can be deleted simultaneously. Similarly to the directional approach, an iteration step of an s-subfield algorithm is composed of s parallel reductions. That is why n-subiteration and n-subfield algorithms are referred to as *n-cycle* algorithms.

In the conventional implementation of thinning algorithm, all border pixels are examined for possible deletion in each thinning phase. Thinning is the fastest skeletonization approach, however, the conventional scheme is a rather time-consuming for objects that also contain ‘thin’ and ‘thick’ parts, since some pixels of the produced centerline ‘get free’ within ‘a few’ (sub)iterations, but the iterative object reduction process is to be continued until no pixels are deleted. That is why the authors of this paper introduced the concept of 1-*attempt thinning* (Palágyi and Németh, 2021a). In a 1-attempt thinning algorithm, if a border pixel cannot be deleted in the very first examination for possible deletion, this pixel remains unchanged in the remaining thinning phases (i.e., it is an element of the produced centerline). In our previous papers we constructed a new fully parallel thinning algorithm, and proved that it is 1-attempt (Palágyi and Németh, 2021a); we managed to prove that an existing 2-subfield parallel thinning algorithm (Németh and Palágyi, 2011) is also 1-attempt (Palágyi and Németh, 2021b); we showed that the thinning algorithms fulfilling the 1-attempt property can be implemented with a remarkable speed up (Palágyi and Németh, 2021a; Palágyi and Németh, 2021b).

In this work, our attention is focussed on 4-cycle parallel thinning: we prove that a 4-subiteration (directional) and a 4-subfield thinning algorithm are 1-

[a] https://orcid.org/0000-0002-3274-7315
[b] https://orcid.org/0000-0002-4118-5019

Palágyi, K. and Németh, G.
1-Attempt 4-Cycle Parallel Thinning Algorithms.
DOI: 10.5220/0010819700003122
In *Proceedings of the 11th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2022)*, pages 229-236
ISBN: 978-989-758-549-4; ISSN: 2184-4313



a b

Figure 1: The two kinds of adjacency relations studied on the square grid (a). The four pixels p_0, p_1, p_2, and p_3 are 4-adjacent to the central pixel p. These four pixels and the four ones marked '•' are 8-adjacent to p.
Partition of $\mathbb{Z}^2$ into four subfields (b). All pixels marked 'i' are in subfield S_i $(i = 0,1,2,3)$.

attempt, and show that their 1-attempt property is useful (i.e., it makes computationally efficient implementation schemes possible).

2 BASIC NOTIONS AND RESULTS

Next, we apply the fundamental concepts of digital topology as reviewed in (Kong and Rosenfeld, 1989).

An $(8,4)$ *picture* on the 2D square grid is a quadruple $(\mathbb{Z}^2, 8, 4, B)$, where an element of $\mathbb{Z}^2$ is assigned to each pixel (i.e., unit square); $B \subseteq \mathbb{Z}^2$ denotes the set of *black pixels*; each pixel in $\mathbb{Z}^2 \setminus B$ is said to be a *white pixel*; adjacency relations 8 and 4 are assigned to B and $\mathbb{Z}^2 \setminus B$, respectively. Two pixels (i.e., unit squares) are 4-*adjacent* if they share an edge, and they are 8-*adjacent* if those pixels share an edge or a vertex, see Fig. 1a. Let $N(p)$ denote the set of (eight) pixels that are 8-adjacent to p.

Since both studied relations are symmetric, their reflexive-transitive closure form equivalence relations, and the generated equivalence classes are called *components*. A *black component* or an *object* is an 8-component of B, while a *white component* is a 4-component of $\mathbb{Z}^2 \setminus B$.

A pixel $p \in B$ is an *interior pixel* for B, if all pixels being 4-adjacent to p are in B, p is called a *border pixel* if it is not an interior pixel, and p is said to be an *isolated pixel* if it forms a singleton object (i.e., it is not 8-adjacent to a black pixel). Here we distinguish four types of border pixels: a black pixel p is called an *i-border pixel* $(i = 0,1,2,3)$ in an $(8,4)$ picture, if the pixel marked p_i in Fig. 1a is white.

Thinning algorithms preserve some (border) pixels that provide relevant geometric information with respect to the shape of the object. Most of existing 2D thinning algorithms preserve *endpixels* (i.e., terminating pixels of curves). The two 4-cycle parallel thinning algorithms described in Section 3 use the following two types of endpixels (Hall, 1996; Németh and Palágyi, 2011):

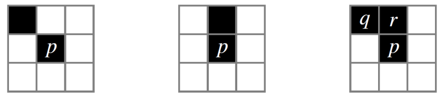


Figure 2: Examples of the studied two types of endpixels. In the first two configurations, p is an endpixel of type 1 and an endpixel of type 2. The third example gives an example of an endpixel of type 2 (that is not an endpixel of type 1).

Definition 1. *A black pixel (in an* $(8,4)$ *picture) is an* endpixel of type 1 *if it is* 8*-adjacent to exactly one black pixel.*

Definition 2. *A black pixel p (in an* $(8,4)$ *picture) is an* endpixel of type 2 *if it is endpixel of type 1 or it is* 8*-adjacent to exactly two black pixels q and r, where q is* 4*-adjacent to r.*

Figure 2 shows some examples of the two types of endpixels taken into consideration.

A crucial issue in thinning algorithms is to ensure *topology preservation* (Kong, 1995; Kong and Rosenfeld, 1989). A reduction in a 2D picture is topology-preserving if each object in the input picture contains exactly one object in the output picture, and each white component in the output picture contains exactly one white component in the input picture. A single black pixel is *simple* for a set of black pixels if its deletion is a topology-preserving reduction. Here we make use of the following characterization of simple pixels:

Theorem 1. (Kong and Rosenfeld, 1989) *A black pixel p is simple in an* $(8,4)$ *picture if all the following conditions hold:*

1. *p is a border pixel.*
2. *There is exactly one black* 8*-component in* $N(p)$.

Note that the simpleness in $(8,4)$ pictures is a local property, it can be decided by examining the eight pixels that are 8-adjacent to the given black pixel, and only non-isolated border pixels may be simple. Here we represent **non**-simple pixels in $(8,4)$ pictures by the six base *matching templates* shown in Fig. 3. All rotated and reflected versions of these base templates match non-simple pixels, too. It can be readily seen that a pixel is non-simple if at least one template matches it.

Parallel reductions can delete a set of black pixels. Hence, we need to consider what is meant by topology preservation when a number of black pixels are deleted simultaneously. Kong gave the following sufficient condition:

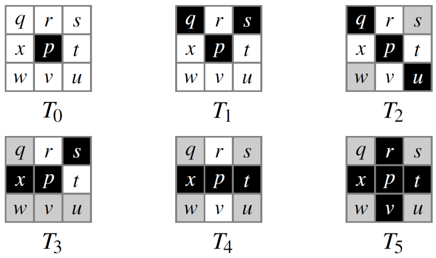


Figure 3: The six base matching templates for characterizing non-simple pixels in $(8,4)$ pictures. Notations: each black template position matches a black pixel; each white element matches a white pixel; each position depicted in gray matches either a black or a white pixel.

Theorem 2. (Kong, 1995) *A parallel reduction acting on* $(8,4)$ *pictures is topology preserving, if all the following conditions hold:*

1. *Only simple pixels are deleted.*
2. *For every set* $\{p,q\}$ *of two* 4*-adjacent pixels that are deleted, p is simple after deletion of q.*
3. *No object contained in a* 2×2 *square is deleted completely.*

Kong's result provides a method for verifying that a fully parallel or a subfield-based thinning algorithm preserves topology. Rosenfeld gave another sufficient condition that is useful in verifying the topological correctness of subiteration-based algorithms:

Theorem 3. (Rosenfeld, 1975) *A parallel reduction acting on* $(8,4)$ *pictures is topology preserving, if all the following conditions hold for every pixel p that is deleted:*

1. *p is an i-border pixel* $(i \in \{0,1,2,3\})$.
2. *p is not an endpixel of type 1.*
3. *There is exactly one black* 8*-component in* $N(p)$.

3 TWO 4-CYCLE PARALLEL THINNING ALGORITHMS

In this section, we describe two 4-cycle topology-preserving parallel thinning algorithm acting on $(8,4)$ pictures on the square grid. The first algorithm **SI** is subiteration-based, and the second one **SF** falls into the category of subfield-based. Algorithm 1 gives the studied two algorithms.

In Algorithm 1, the kernel of the **repeat** cycle corresponds to one iteration step that comprises four subiterations (i.e., four parallel reductions). In these subiterations, algorithm **SI** considers the four deletion directions that are specified by the studied four types of border pixels, and algorithm **SF** alternatively activates the four subfields shown in Fig. 1b.

Algorithm 1: 4-cycle algorithm **T** (**T**=**SI**,**SF**).

Input: picture $(\mathbb{Z}^2, 8, 4, X)$
Output: picture $(\mathbb{Z}^2, 8, 4, Y)$
$Y \leftarrow X$
repeat
  `// one iteration step`
  for $i \leftarrow 0$ **to** 3 **do**
    `// i-th subiteration`
    $D(i) \leftarrow \{\, p \mid p$ is **T**-i-deletable for $Y \,\}$
    $Y \leftarrow Y \setminus D(i)$
until $D(0) \cup D(1) \cup D(2) \cup D(3) = \emptyset$;

The description of the two 4-cycle parallel thinning algorithms (i.e., algorithms **SI** and **SF**) is completed by specifying their deletion rules:

Definition 3. *A black pixel p in picture* $(\mathbb{Z}^2, 8, 4, B)$ *is* **SI**-*i*-deletable *for B* $(i = 0,1,2,3)$ *if all the following conditions hold:*

1. *p is an i-border pixel.*
2. *p is not an endpixel of type 2.*
3. *p is simple.*

A black pixel (in an $(8,4)$ picture) is *non-***SI**-*i-deletable* if it is not **SI**-*i*-deletable $(i = 0,1,2,3)$, i.e., at least one condition of Definition 3 is violated.

Definition 4. *A black pixel p in picture* $(\mathbb{Z}^2, 8, 4, B)$ *is* **SF**-*i*-deletable *for B* $(i = 0,1,2,3)$ *if all the following conditions hold:*

1. $p \in S_i$ *(i.e., it is in the i-th subfield, see Fig. 1b).*
2. *p is not an endpixel of type 1.*
3. *p is simple.*

A black pixel (in an $(8,4)$ picture) is said to be a *non-***SF**-*i-deletable* pixel if it is not **SF**-*i*-deletable $(i = 0,1,2,3)$, i.e., at least one condition of Definition 4 is not satisfied.

Since all endpixels of type 1 are also endpixels of type 2, the following statement holds:

Proposition 1. *All endpixels of type 1 are not* **SI**-*i-deletable* $(i = 0,1,2,3)$.

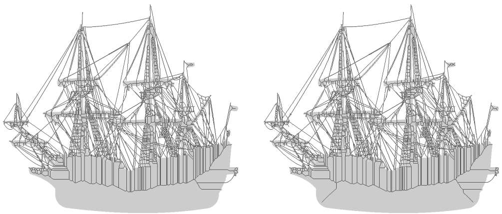
Let us state another obvious proposition:

Proposition 2. *If* $p \in S_i$ $(i = 0,1,2,3)$ *and* $q \in N(p)$, $q \notin S_i$.

By Proposition 1 and Proposition 2, it can be readily seen that the parallel reductions that delete **SI**-*i*-deletable or **SF**-*i*-deletable pixels satisfy all conditions of Theorem 3. Thus both algorithms are topology-preserving.

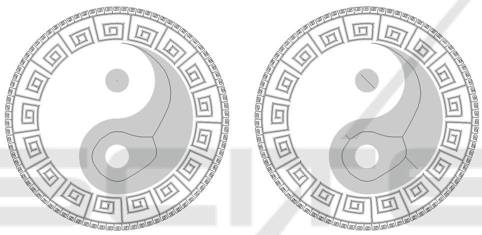
Note that the idea of algorithm **SI** is originated from Rosenfeld, but his early algorithm preserves endpixels of type 1 (Rosenfeld, 1975). Algorithm **SF** is proposed in (Németh and Palágyi, 2011), and it is referred to as **SF-4-1**.

For reasons of scope, we do not analyze the geometric properties of the studied two 4-cycle algorithms. Here we present five illustrative examples of the centerlines produced by algorithms **SI** and **SF**, see Fig. 4-8. The pairs of numbers in parentheses are the counts of object pixels in the produced centerlines and the number of the required iterations.



SI (39685, 107) **SF** (39824, 107)

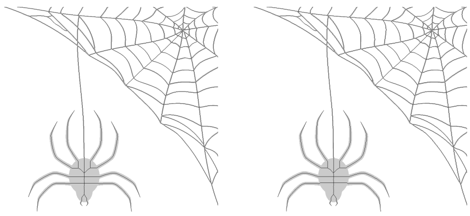
Figure 4: Centerlines produced by the studied algorithms superimposed on a 730×820 image of a ship. The original image contains 214310 object pixels.



SI (17128, 123) **SF** (17366, 123)

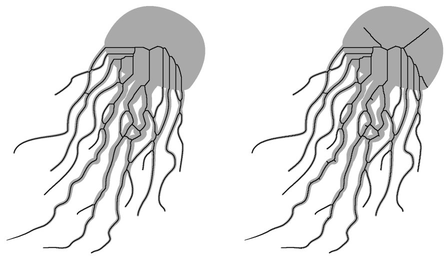
Figure 5: Centerlines produced by the studied algorithms superimposed on a 730×730 image of the yin and yang symbol. The original image contains 190246 object pixels.

By Figs. 4-8, we can state that both algorithms **SI** and **SF** produce one pixel width and well-positioned centerlines.



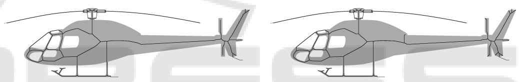
SI (10092, 61) **SF** (10104, 61)

Figure 6: Centerlines produced by the studied algorithms superimposed on a 730×795 image of a spider. The original image contains 58557 object pixels.



SI (7555, 175) **SF** (7839, 175)

Figure 7: Centerlines produced by the studied algorithms superimposed on a 990×804 image of a jellyfish. The original image contains 214514 object pixels.



SI (7652, 186) **SF** (7839, 186)

Figure 8: Centerlines produced by the studied algorithms superimposed on a 640×2200 image of a helicopter. The original image contains 399984 object pixels.

a b

Figure 9: Templates (configurations) associated with Lemma 1 (a) and Lemma 2 (b).

4 FULFILLING THE 1-ATTEMPT PROPERTY

In this section, we show that the studied algorithms **SI** and **SF** fulfill the 1-attempt property.

In order to prove that both algorithms are 1-attempt, we shall state two lemmas, introduce a new concept, and examine the matching templates shown in Fig. 9.

Lemma 1. *Let p be a black pixel (in an $(8,4)$ picture) that is matched by the template shown in Fig. 9a (or its rotated versions). Then p cannot be deleted by algorithms* **SI** *and* **SF**.

Proof. By Definition 3, Definition 4, and Proposition 1, only simple pixels that are not endpixels of type 1 can be deleted by algorithms **SI** and **SF**.

Since $\{q\}$ is a black 8-component in $N(p)$, pixel p is simple (by Theorem 1) iff all the five pixels a, b, c, d, and e in $N(p) \setminus \{q\}$ are white. In this case p is an endpixel of type 1. Thus p cannot be deleted by algorithms **SI** and **SF**. □

Lemma 2. *Let p be a black pixel (in an $(8,4)$ picture) that is matched by the template shown in Fig. 9b (or its rotated versions). Then p cannot be deleted by algorithms* **SI** *and* **SF**.

Proof. By Definition 3, Definition 4, and Proposition 1, only simple pixels that are not endpixels of type 1 can be deleted by algorithms **SI** and **SF**.

Since $\{q\}$ is a singleton black 8-component in $N(p)$, pixel p is simple (by Theorem 1) iff all the three pixels a, b, and c in $N(p) \setminus \{q\}$ are white. In this case

p is an endpixel of type 1. Thus p cannot be deleted by algorithms **SI** and **SF**. $\square$

Now let us introduce a new concept.

Definition 5. *Let $p \in B$ be a non-simple pixel in picture $(\mathbb{Z}^2, 8, 4, B)$. The set of pixels $S \subseteq N(p) \cap B$ is a* simplifier set *(associated to p) if pixel p is simple in picture $(\mathbb{Z}^2, 8, 4, B \setminus S)$.*

For the sake of brevity in the following we will refer to positions in the matching templates shown in Fig. 3 as pixels in $(8,4)$ pictures.

Proposition 3. *Let p be a non-simple border (black) pixel (in an $(8,4)$ picture) that is matched by template T_0 (see Fig. 3). Then p cannot be deleted by algorithms* **SI** *and* **SF**.

Proof. Since p is an isolated pixel (i.e., there is no black pixel in $N(p)$), there is no simplifier set associated to p. Thus p remains a non-simple black pixel.

By Definition 3 and Definition 4, only simple pixels can be deleted by algorithms **SI** and **SF**. Thus p cannot be deleted. $\square$

Proposition 4. *Let $p \in B$ be a non-simple border (black) pixel in picture $(\mathbb{Z}^2, 8, 4, B)$ that is matched by template T_1 (see Fig. 3). Then p cannot be deleted by algorithms* **SI** *and* **SF**.

Proof. We can state that $\{q\}$ and $\{s\}$ are the two possible simplifier sets associated to p. This follows from a careful examination of T_1.

Since, by Lemma 1, q and s cannot be deleted by algorithms **SI** and **SF**, p remains a non-simple black pixel.

By Definition 3 and Definition 4, only simple pixels can be deleted by the studied two algorithms. Thus p cannot be deleted. $\square$

Proposition 5. *Let us assume that a non-simple border (black) pixel p (in an $(8,4)$ picture) is matched by template T_2 (see Fig. 3). Then p cannot be deleted by algorithms* **SI** *and* **SF**.

Proof. Since, by Lemma 1, all the two black pixels (i.e., q and u) and the two potentially black ones (i.e., s and w) cannot be deleted by algorithms **SI** and **SF**, p remains a non-simple black pixel.

By Definition 3 and Definition 4, only simple pixels can be deleted by algorithms **SI** and **SF**. Thus p cannot be deleted. $\square$

Proposition 6. *Let us assume that a non-simple border (black) pixel $p \in B$ in picture $(\mathbb{Z}^2, 8, 4, B)$ is matched by template T_3 (see Fig. 3). Then p cannot be deleted by algorithms* **SI** *and* **SF**.

Proof. It can be readily seen that $\{s\}$ and $S = \{q,u,v,w,x\} \cap B$ are the two possible simplifier sets associated to p.

Since, by Lemma 1, pixel s cannot be deleted in the thinning process of none of the two algorithms **SI** and **SF**, let us examine the deletability of S.

It is obvious that p becomes an endpixel of type 1 after the deletion of S.

By Definition 3 and Definition 4, only simple pixels that are not endpixels of type 1 can be deleted by algorithms **SI** and **SF**. Thus the simplified p cannot be deleted. $\square$

Let us summarize the previously stated four propositions:

Proposition 7. *If a non-simple border (black) pixel p is matched by template T_0, T_1, T_2, or T_3 in the input or an interim picture of algorithms* **SI** *or* **SF**, *p cannot be deleted in the remaining thinning phases (i.e., it is an element of the produced centerline).*

Let us take the remaining two matching templates shown in Fig. 3 into consideration.

Proposition 8. *Let $p \in B$ be a non-simple border (black) pixel in picture $(\mathbb{Z}^2, 8, 4, B)$ that is matched by template T_4 (see Fig. 3). Then exactly one of the two sets of pixels $\{q,w,x\} \cap B$ and $\{s,t,u\} \cap B$ is a subset of all possible simplifier sets associated to p.*

This simply follows from Theorem 1 and a careful examination of template T_4.

Lastly an absolutely obvious statement is made:

Proposition 9. *Let us assume that a (non-simple black) pixel p (in an $(8,4)$ picture) is matched by template T_5 (see Fig. 3). Then p is not a border pixel (i.e., it is an interior pixel).*

4.1 Examining Algorithm SI

We are now ready to state the following theorem:

Theorem 4. *Algorithm* **SI** *is* 1*-attempt.*

Proof. Without loss of generality, we can assume that 0-border pixels are examined for possible deletion in the actual subiteration.

We need to verify that if a 0-border pixel $p \in B$ in picture $(\mathbb{Z}^2, 8, 4, B)$ is non-**SI**-0-deletable in the actual subiteration, it remains non-**SI**-i-deletable in the remaining thinning phases for any $i \in \{0,1,2,3\}$.

Let us assume indirectly that p is **SI**-i-deletable for $B \setminus D$ for some $i \in \{0,1,2,3\}$ and for some set of deleted pixels $D \subseteq B \setminus \{p\}$.

Since pixel p is a non-**SI**-0-deletable (0-border) pixel for B, at least one of the last two conditions of Definition 3 is violated. Consequently,

- p is an endpixel of type 2 for B, or
- p is non-simple for B.

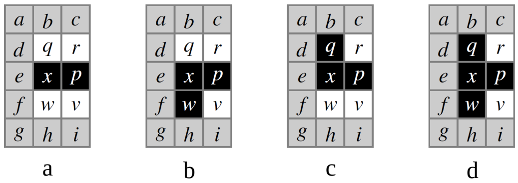


Figure 10: Configurations associated with Theorem 4 and Theorem 5.

It is obvious that if p is an endpixel of type 2 for B (i.e., the first point holds), p remains an endpixel of type 2 for $B \setminus D$ for any $D \subseteq B \setminus \{p\}$. Thus only the second point is to be examined (i.e., p is non-simple for B).

According to our assumption, p is deleted in a remaining thinning phase, thus it becomes simple for $B \setminus D$. It means that $N(p) \cap D$ is a simplifier set of deleted pixels.

Since p is a non-simple pixel for B, at least one template shown in Fig. 3 matches it. By Proposition 7 and Proposition 9 we can ignore the five templates T_0, T_1, T_2, T_3, and T_5. Thus template T_4 is the only one to be investigated.

If p is matched by T_4, by Proposition 8, exactly one of the two sets of pixels $\{q,w,x\} \cap B$ and $\{s,t,u\} \cap B$ is a subset of all possible simplifier sets associated to p. Consequently, one of $\{q,w,x\} \cap B$ and $\{s,t,u\} \cap B$ is to be completely deleted. Without loss of generality, we can assume that $(\{q,w,x\} \cap B) \subseteq D$. It is known that black pixel $x \in D$ (see Fig. 3), thus the following four cases (shown in Fig. 10) are to be checked:

- $q \notin D, w \notin D$ (see Fig. 10a):
  In this case, by Lemma 2, pixel x cannot be deleted. This is a contradiction as $x \in D$.
- $q \notin D, w \in D$ (see Fig. 10b):
  Then the following points have to be checked:
  - If x and w are deleted in different subiterations, and x is deleted first, then, by Lemma 1, pixel w cannot be deleted. This is a contradiction as $w \in D$.
  - If x and w are deleted in different subiterations, and w is deleted first, then, by Lemma 2, pixel x cannot be deleted. This is a contradiction as $x \in D$.
  - If x and w are deleted simultaneously (i.e., in the same subiteration), then
    - w is not **SI**-0-deletable, since it is not a 0-border pixel (i.e., pixel x is black);
    - x is not **SI**-1-deletable, since it is not a 1-border pixel (i.e., pixel p is black);
    - x is not **SI**-2-deletable, since it is not a 2-border pixel (i.e., pixel w is black).

    Thus both pixels x and w need to be **SI**-3-deletable. Consequently, the two pixels e and f are white. It can be readily seen by Theorem 1 that pixel x is simple iff pixel d is also white. In this case, x is an endpixel of type 2. Thus x is not **SI**-3-deletable, and we arrive at a contradiction.
- $q \in D, w \notin D$ (see Fig. 10c):
  Similarly to the previous case, the following three points are to be examined:
  - If x and q are deleted in different subiterations, and x is deleted first, then, by Lemma 1, pixel q cannot be deleted. This is a contradiction as $q \in D$.
  - If x and q are deleted in different subiterations, and q is deleted first, then, by Lemma 2, pixel x cannot be deleted. This is a contradiction as $x \in D$.
  - If x and q are deleted simultaneously (i.e., in the same subiteration), then
    - x is not **SI**-0-deletable, since it is not a 0-border pixel (i.e., pixel q is black);
    - x is not **SI**-1-deletable, since it is not a 1-border pixel (i.e., pixel p is black);
    - q is not **SI**-2-deletable, since it is not a 2-border pixel (i.e., pixel x is black).

    Thus both pixels x and w need to be **SI**-3-deletable. Consequently, the two pixels d and e are white. It can be readily seen by Theorem 1 that pixel x is simple iff pixel f is also white. In this case, x is an endpixel of type 2. Thus x is not **SI**-3-deletable, and we arrive at a contradiction.
- $q \in D, w \in D$ (see Fig. 10d):
  In this case, the following two points are to be checked:
  - Let us assume that the three pixels q, x, and w are deleted in different subiterations.
    - If just one of the two pixels q and w is deleted first, then we get the previously examined cases shown in Fig. 10b and Fig. 10c. Thus we arrive at a contradiction.
    - If pixel x is deleted first, then both pixels q and w cannot be deleted by Lemma 1. This is a contradiction as $q \in D$ and $w \in D$.
    - If the two pixels q and w are deleted first, then we get the previously examined case shown in Fig. 10a. Thus we arrive at a contradiction.
    - If the two pixels q and x are deleted first, then w cannot be deleted by Lemma 1. This is a contradiction as $w \in D$.

  * If the two pixels x and w are deleted first, then q cannot be deleted by Lemma 1. This is a contradiction as $q \in D$.
- Lastly, let us assume that the three pixels q, x, and w are deleted at a time (i.e., in the same subiteration). In this case
  * x is not **SI**-0-deletable, since it is not a 0-border pixel (i.e., pixel q is black);
  * x is not **SI**-1-deletable, since it is not a 1-border pixel (i.e., pixel p is black);
  * x is not **SI**-2-deletable, since it is not a 2-border pixel (i.e., pixel w is black).

  Thus all the three pixels q, x, and w need to be **SI**-3-deletable. Consequently, pixels d, e, and f are all white. It can be readily seen, by Theorem 1, pixel q is simple iff all the three pixels a, b, and c are also white. In this case, q is an endpixel of type 2. Thus q is not **SI**-3-deletable, and we arrive at a contradiction. (Similarly, by Theorem 1, pixel w is simple iff all the three pixels g, h, and i are also white. Since w is an endpixel of type 2, it is not **SI**-3-deletable. Thus we arrive at a contradiction.)

Since a non-**SI**-0-deletable border pixel may not be **SI**-i-deletable $(i = 0, 1, 2, 3)$ in the remaining subiterations of the thinning process, this theorem holds. □

4.2 Examining Algorithm SF

Similarly to algorithm **SI**, we can state that algorithm **SF** fulfills the 1-attempt property:

Theorem 5. *Algorithm* **SF** *is* 1*-attempt.*

Proof. Without loss of generality, we can assume that a border pixel p to be examined is in S_0.

We need to verify that if $p \in B$ in picture $(\mathbb{Z}^2, 8, 4, B)$ is non-**SF**-0-deletable in the actual subiteration, it remains non-**SF**-0-deletable.

Let us assume indirectly that p is **SF**-0-deletable for $B \setminus D$, where $D \subseteq B \setminus \{p\}$ is a set of deleted pixels.

Since pixel p is a non-**SF**-0-deletable border pixel for B, at least one of the last two conditions of Definition 4 is violated. Consequently,

- p is an endpixel of type 1 for B, or
- p is non-simple for B.

It is obvious that if p is an endpixel of type 1 for B (i.e., the first point holds), p remains an endpixel of type 1 for $B \setminus D$ for any $D \subseteq B \setminus \{p\}$. Thus only the second point is to be examined (i.e., p is non-simple for B).

According to our assumption, p is deleted in a remaining thinning phase, thus it becomes simple for $B \setminus D$. It means that $N(p) \cap D$ is a simplifier set of deleted pixels.

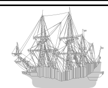
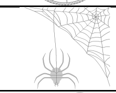
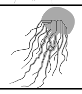
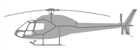
Since p is a non-simple pixel for B, at least one template shown in Fig. 3 matches it. By Proposition 7 and Proposition 9 we can ignore the five templates T_0, T_1, T_2, T_3, and T_5. Thus T_4 is the only template to be investigated.

If p is matched by T_4, by Proposition 8, exactly one of the two sets of pixels $\{q,w,x\} \cap B$ and $\{s,t,u\} \cap B$ is a subset of all possible simplifier sets associated to p. Consequently, one of $\{q,w,x\} \cap B$ and $\{s,t,u\} \cap B$ is to be completely deleted. Without loss of generality, we can assume that $(\{q,w,x\} \cap B) \subseteq D$. It is known that black pixel $x \in D$ (see Fig. 3), thus the following four cases depicted in Fig. 10 are to be checked:

- $q \notin D, w \notin D$ (see Fig. 10a):
  By Lemma 2, pixel x cannot be deleted. This is a contradiction as $x \in D$.
- $q \notin D, w \in D$ (see Fig. 10b):
  By Proposition 2, x and w belong to different subfields. Thus these two pixels cannot be deleted in the same subiteration.
  - If x is deleted first, w cannot be deleted by Lemma 1. This is a contradiction as $w \in D$.
  - If w is deleted first, x cannot be deleted by Lemma 2. This is a contradiction as $x \in D$.
- $q \in D, w \notin D$ (see Fig. 10c):
  Similarly to the previous case, by Proposition 2, x and q belong to different subfields. Thus these two pixels cannot be deleted in the same subiteration.
  - If x is deleted first, q cannot be deleted by Lemma 1. This is a contradiction as $q \in D$.
  - If q is deleted first, x cannot be deleted by Lemma 2. This is a contradiction as $x \in D$.
- $q \in D, w \in D$ (see Fig. 10d):
  By Proposition 2, x and q are in different subfields, x and w are also in different subfields. (Note that q and w are in the same subfield, see Fig. 1b.) Then the following points are to be checked:
  - If just one of the two pixels q and w is deleted first, then we get the previously examined cases shown in Fig. 10b and Fig. 10c, respectively. Thus we arrive at a contradiction.
  - If x is deleted first, then both pixels q and w cannot be deleted by Lemma 1. This is a contradiction as $q \in D$ and $w \in D$.

Since a non-**SF**-0-deletable border pixel in S_0 may not be **SF**-i-deletable $(i = 0, 1, 2, 3)$ in the remaining subiterations, the proof by contradiction is completed. □

Table 1: Speed up $\mathbf{C}/\mathbf{A}$ for the five test images shown in Fig. 4-8 concerning the two 1-attempt algorithms, where **C** is the computation time (in sec.) at the 'conventional' scheme, and **A** is the required runing time according to the advanced '1-attempt' implementation.

Test image	Speed-up	
	algorithm **SI**	algorithm **SF**
	$\frac{1.305}{0.046} = \mathbf{28.37}$	$\frac{0.468}{0.037} = \mathbf{12.65}$
	$\frac{0.720}{0.036} = \mathbf{20.00}$	$\frac{0.264}{0.032} = \mathbf{8.25}$
	$\frac{0.182}{0.011} = \mathbf{16.55}$	$\frac{0.067}{0.010} = \mathbf{6.70}$
	$\frac{0.374}{0.038} = \mathbf{9.84}$	$\frac{0.166}{0.037} = \mathbf{4.49}$
	$\frac{0.403}{0.067} = \mathbf{6.01}$	$\frac{0.208}{0.061} = \mathbf{3.41}$

5 USEFULNESS OF THE 1-ATTEMPT PROPERTY

A general and computationally efficient implementation scheme for parallel thinning algorithms was proposed in (Palágyi, 2008). This 'conventional' scheme takes advantage of the fact that all thinning algorithms may delete only border pixels. Thus we do not have to examine the deletability of interior pixels, and the repeated scans of the entire array (that stores the actual finite picture) can be avoided by using a linked list that stores the border pixels that are present in the input picture of the actual thinning phase.

In our previous papers, we proposed 'advanced' implementation for 1-attempt thinning (Palágyi and Németh, 2021a; Palágyi and Németh, 2021b), and we gained remarkable speed up over the 'conventional' scheme.

The usefulness of the 1-attempt property of the studied two 4-cycle algorithms **SI** and **SF** is demonstrated by Table 1.

Both implementation schemes under comparison were run on a usual laptop (Asus VivoBook S14; 1.8 GHz Intel Core i7; Fedora 32 Cinnamon, 64-bit) and written in C++. Note that just the iterative thinning process itself was considered here; reading the input image and writing the output image were not taken into account.

We strongly emphasize that the attainable speed-up is quite image-dependent. Notice that algorithm **SI**

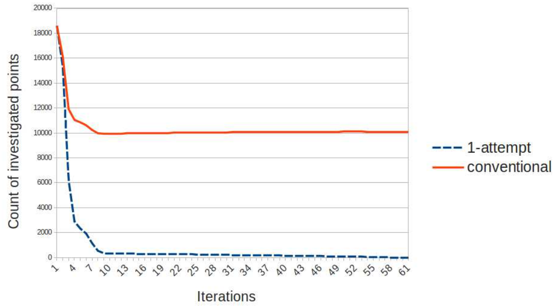


Figure 11: Comparison of the behavior of the 'conventional' and the '1-attempt' implementations of algorithm **SI** for the image of a spider (see Fig. 6). While the 'conventional' scheme evaluates the same border pixel several times, the '1-attempt' approach investigates each pixel just once.

results in a larger speed-up than the achievable speed-up of algorithm **SF**. This is why a border pixel my be a 'morefold' border pixel but each pixel is in just one subfield. Figure 11 illustrates the differences between the 'conventional' and the '1-attempt' implementations.

REFERENCES

Hall, R. W. (1996). Parallel connectivity-preserving thinning algorithms. In Kong, T. Y. and Rosenfeld, A., editors, *Topological algorithms for digital image processing*, pages 145–179. Elsevier Science.

Kong, T. Y. (1995). On topology preservation in 2-d and 3-d thinning. *Int. J. Pattern Recognit. Artif. Intell.*, 9:813–844.

Kong, T. Y. and Rosenfeld, A. (1989). Digital topology: Introduction and survey. *Comput. Vis. Graph. Image Process.*, 48:357–393.

Németh, G. and Palágyi, K. (2011). Topology preserving parallel thinning algorithms. *Int. J. Imaging Syst. Technol.*, 21:37–44.

Palágyi, K. (2008). A 3d fully parallel surface-thinning algorithm. *Theor. Comput. Sci.*, 406:119–135.

Palágyi, K. and Németh, G. (2021a). 1-attempt parallel thinning. *J. Comb. Optim. (in press)*.

Palágyi, K. and Németh, G. (2021b). 1-attempt subfield-based parallel thinning. In *Proc. of the 12th Int. Symposium on Image and Signal Processing and Analysis, ISPA 2021*, pages 219–224.

Rosenfeld, A. (1975). A characterization of parallel thinning algorithms. *Inf. Control*, 29:286–291.

Saha, P. K., Borgefors, G., and Sanniti di Baja, G., editors (2017). *Skeletonization: Theory, methods and applications*. Academic Press.

Siddiqi, K. and Pizer, S., editors (2008). *Medial Representations – Mathematics, Algorithms and Applications, Computational Imaging and Vision, vol. 37*. Springer, New York.