# An order-independent sequential thinning algorithm

Péter Kardos
University of Szeged

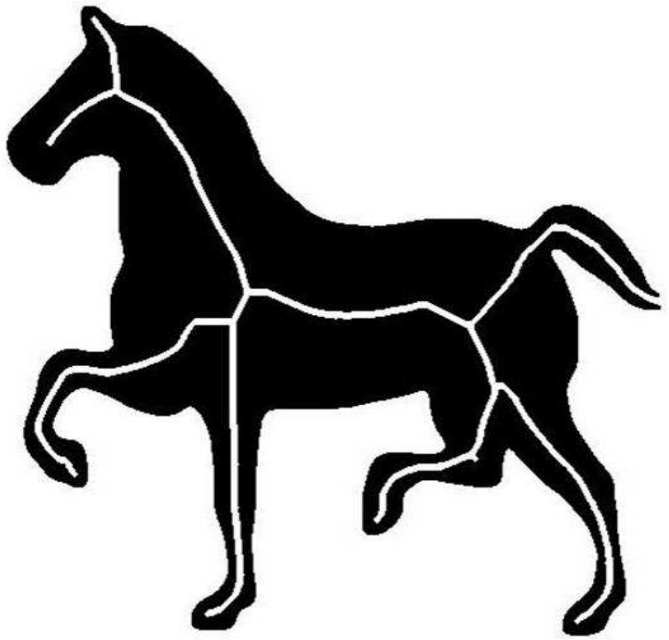
*Summer School on Image Processing 2009*

# Content

- Skeletonization by thinning
- Order-independent thinning
- Proposed algorithm
- Results

# Skeleton

- Region-based shape descriptor
- Definitions:
  - Result of the Medial Axis Transform
  - Praire-fire analogy
  - Centers of maximal inscribed hyperspheres.

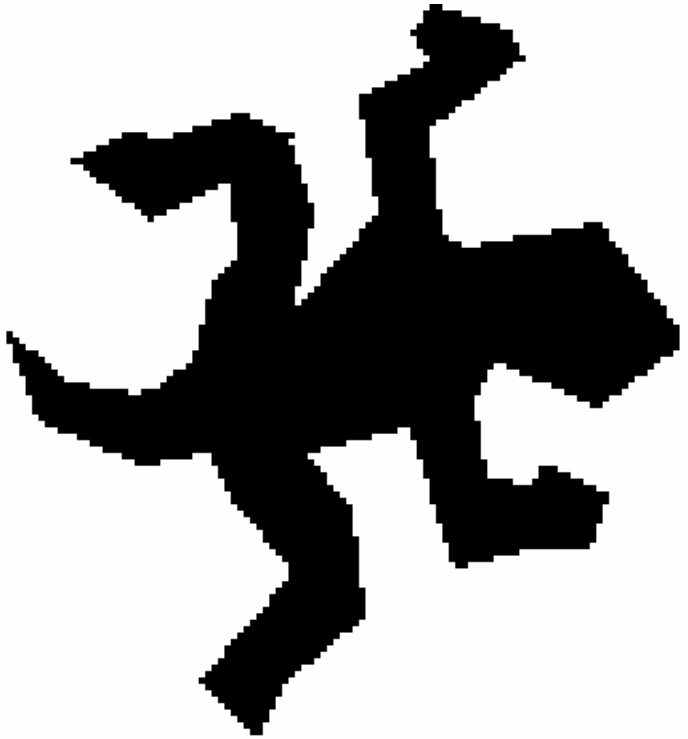


# Requirements of skeletonization

- Topology preserving
- 1-pixel thin
- Medialness
- Invariant under the most important geometrical transformations

# Thinning

- Simulation of the fire-front propagation
- Iterative object reduction
- Parallel and sequential alternatives
  - In sequential case the skeleton depends on the visiting order of border points



# Sequential thinning algorithms

**repeat**

*Phase 1:*

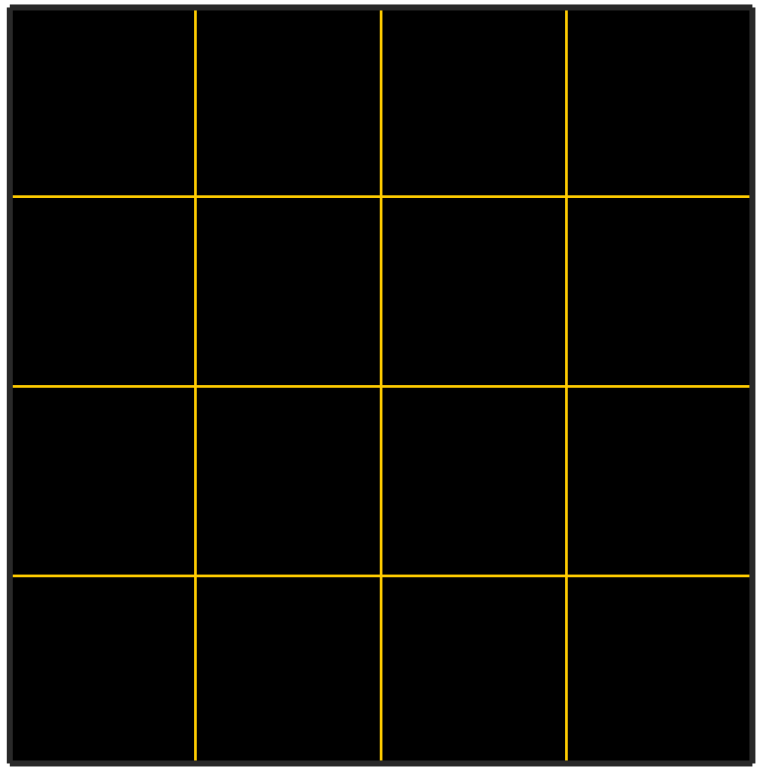
label border points in the image

*Phase 2:*

remove deletable border points

**until** no points are deleted

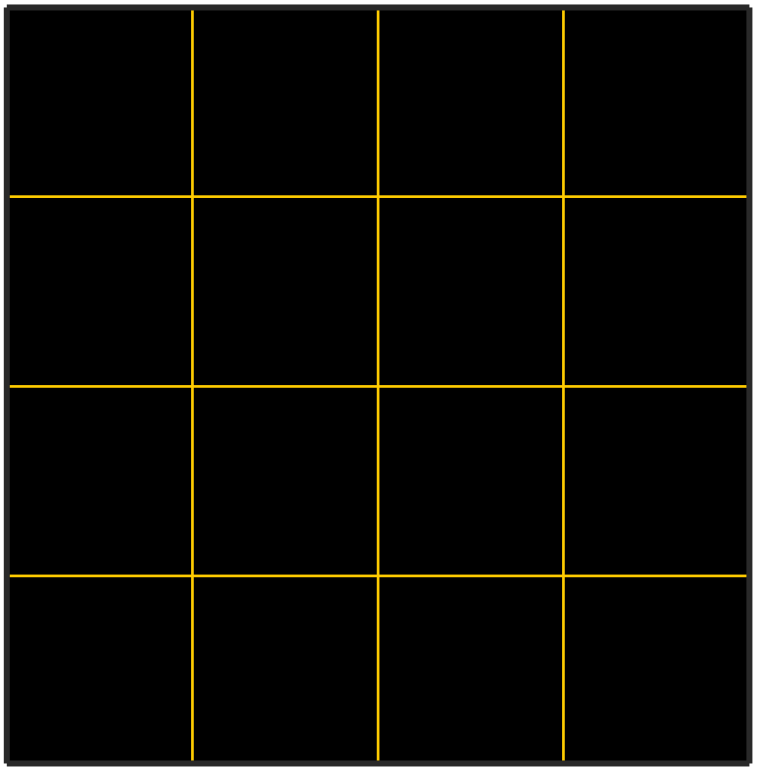
# Problem of order-independency



# Problem of order-independency

*forward scan*

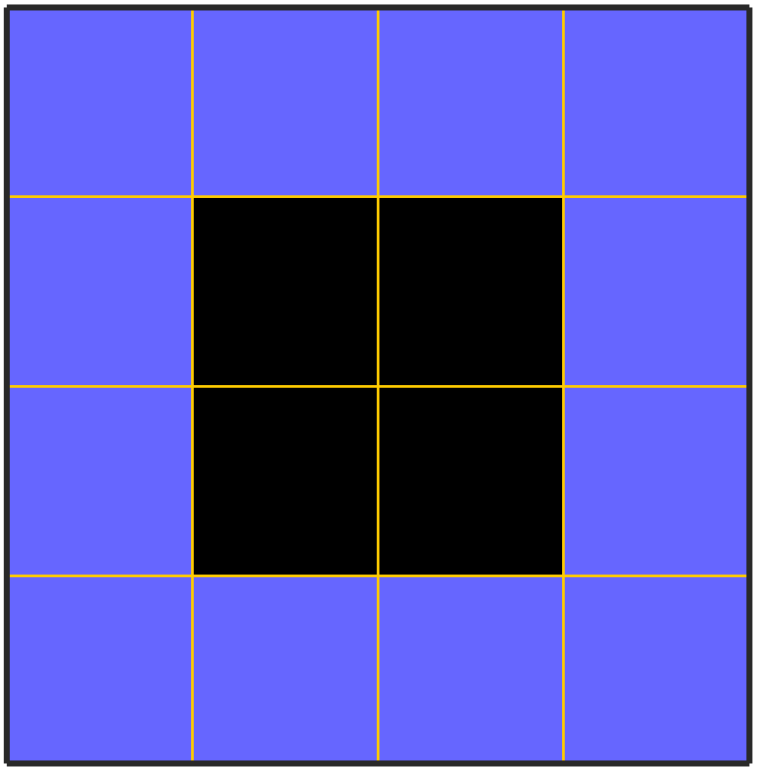
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

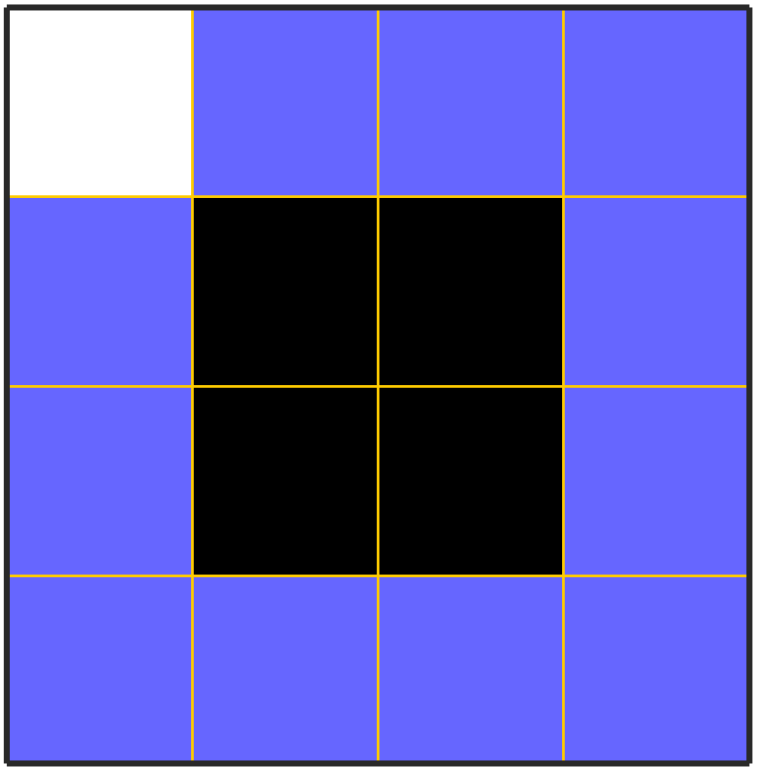
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

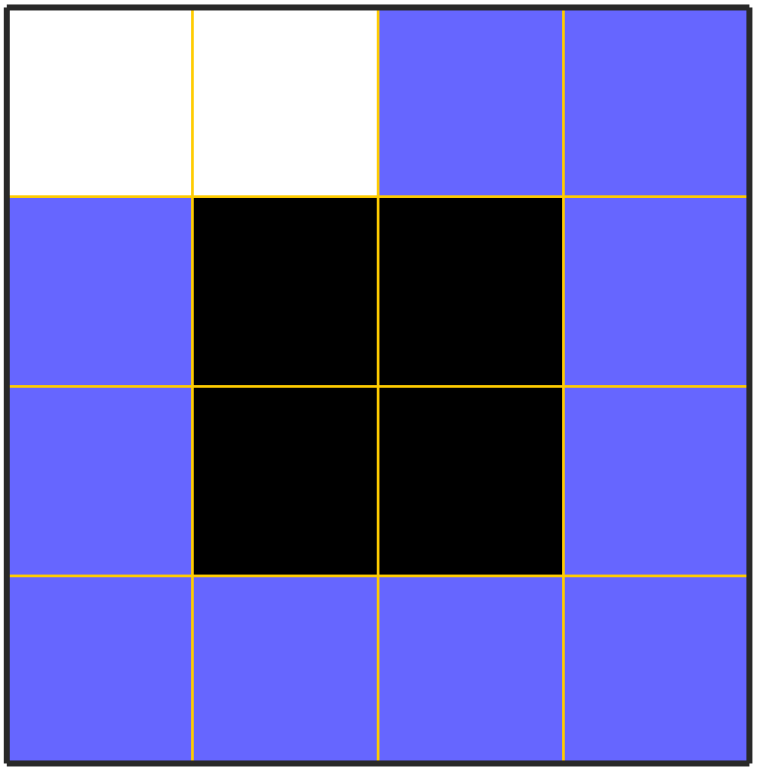
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

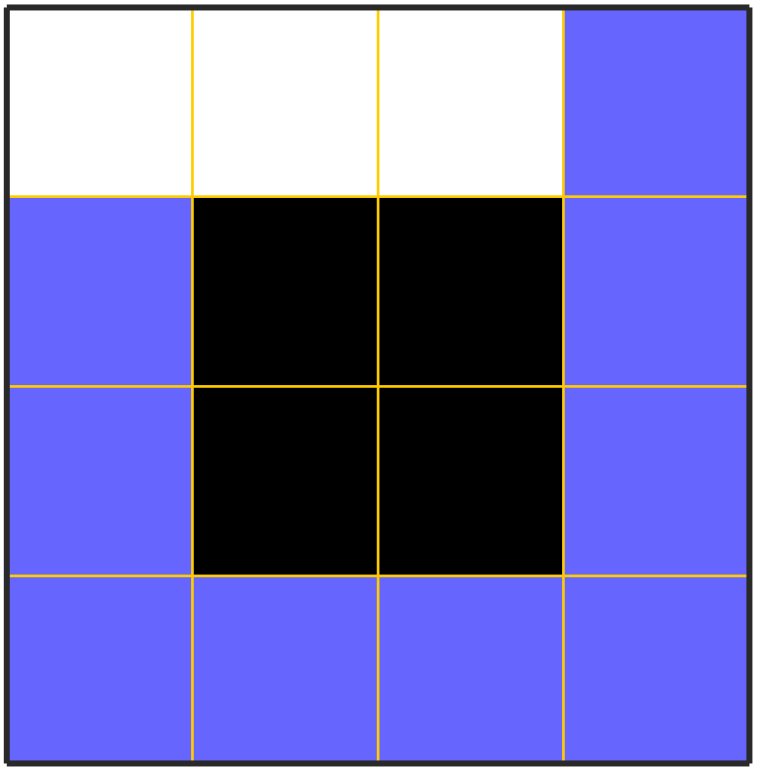
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

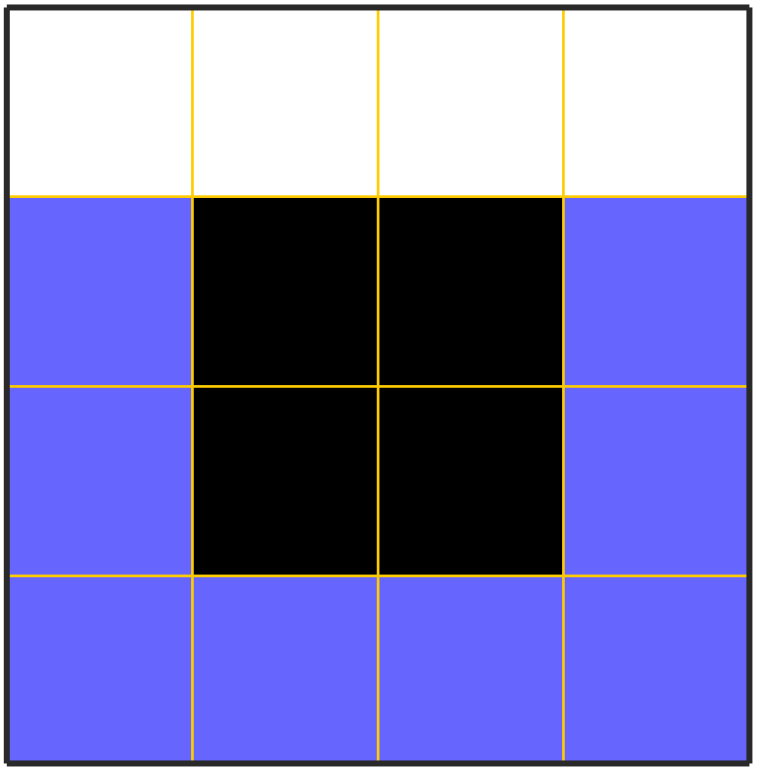
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

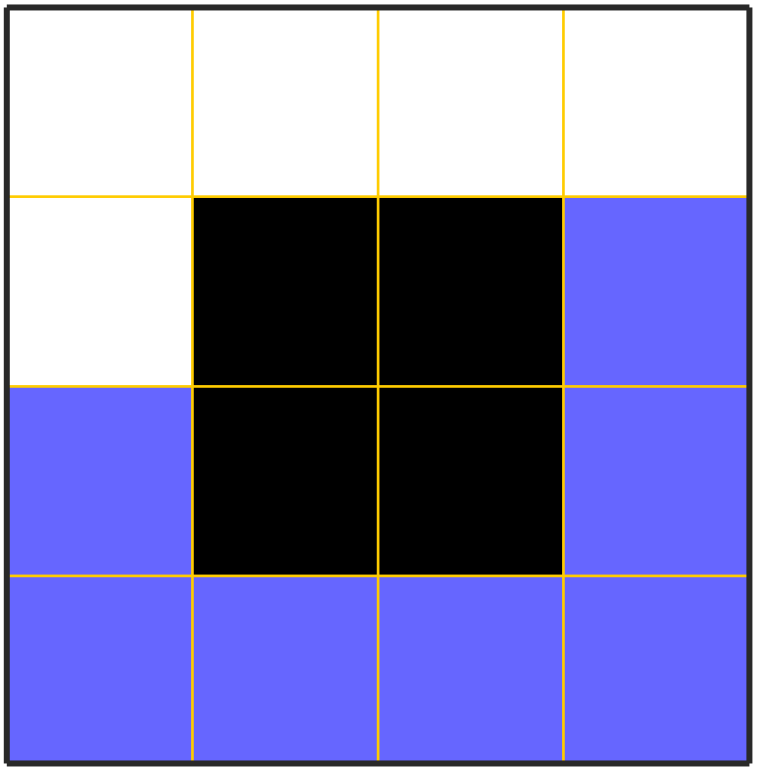
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

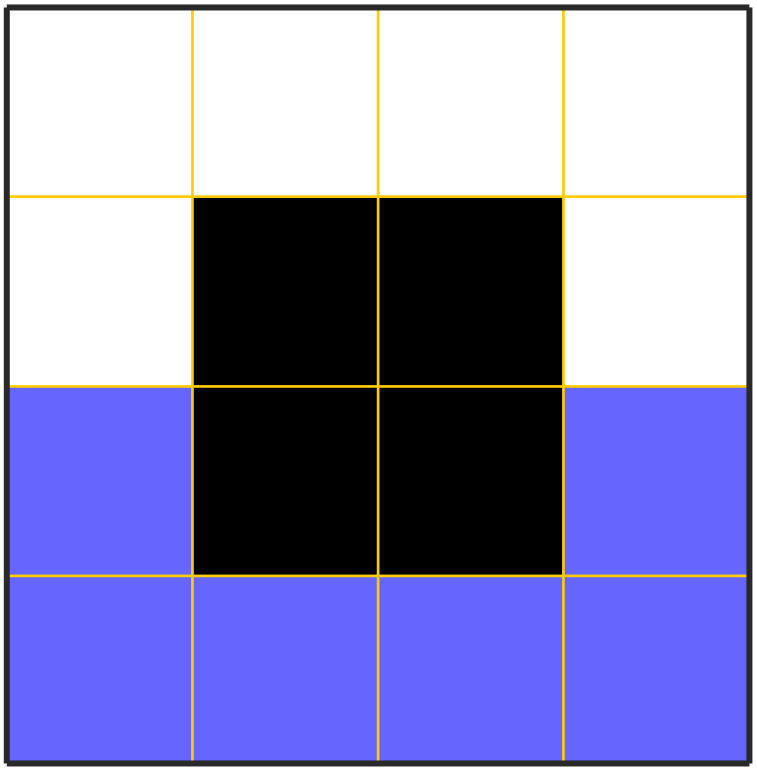
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

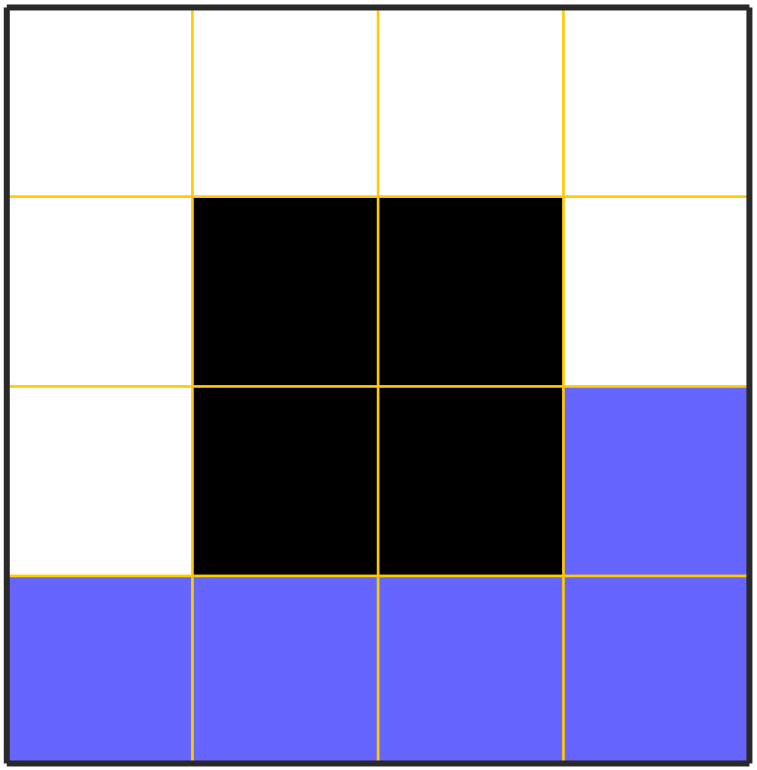
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

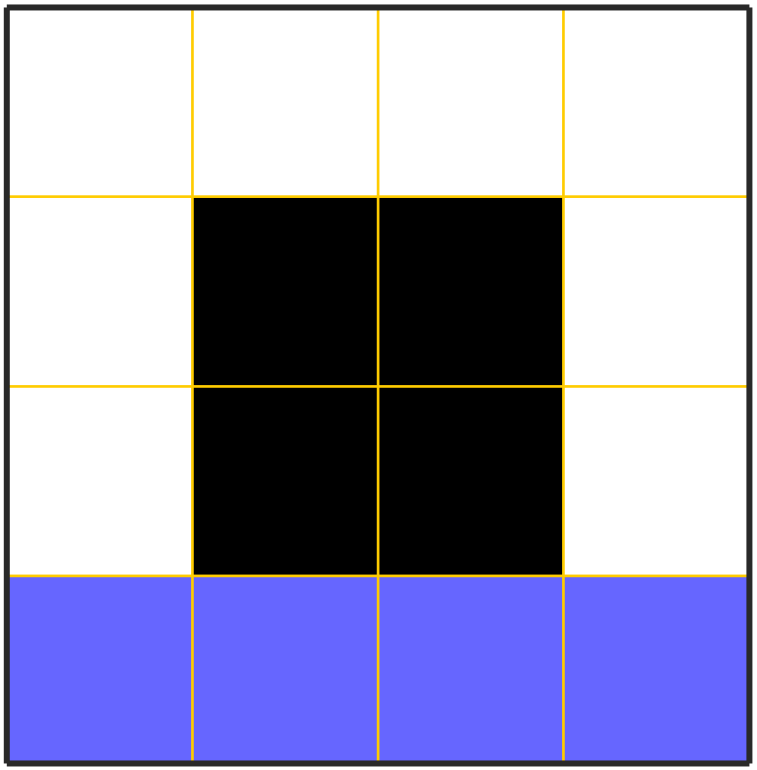
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

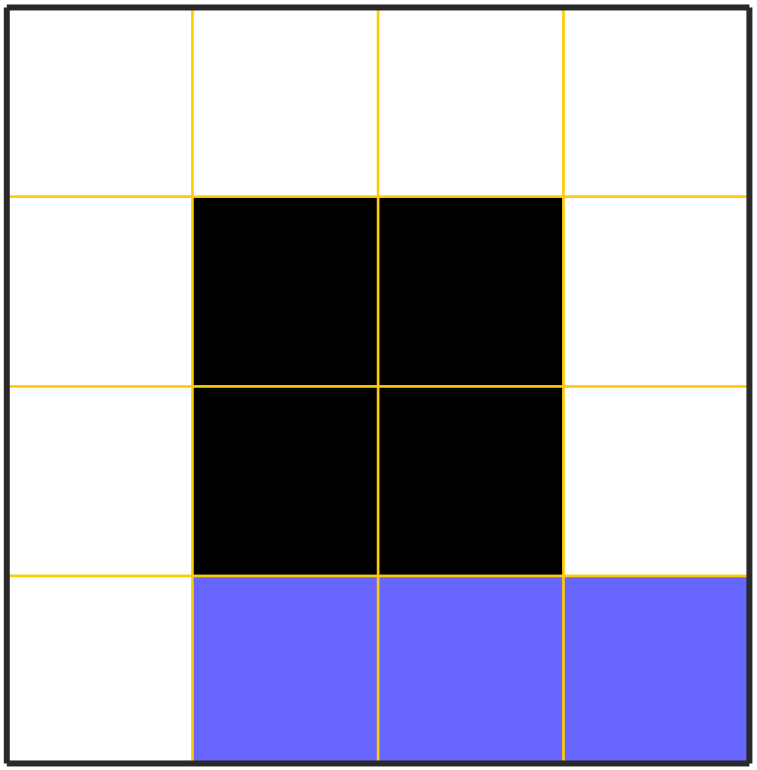
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

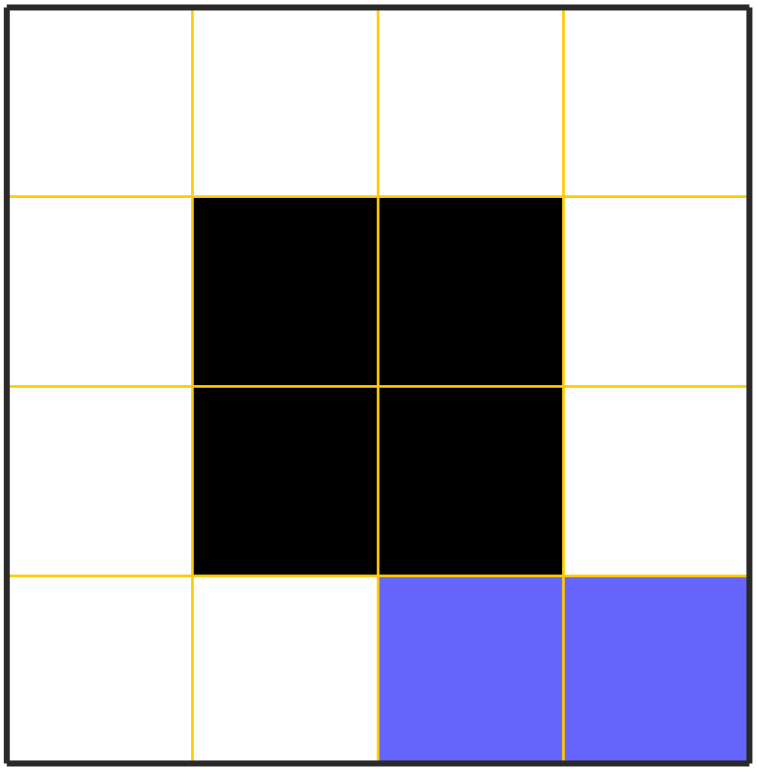
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

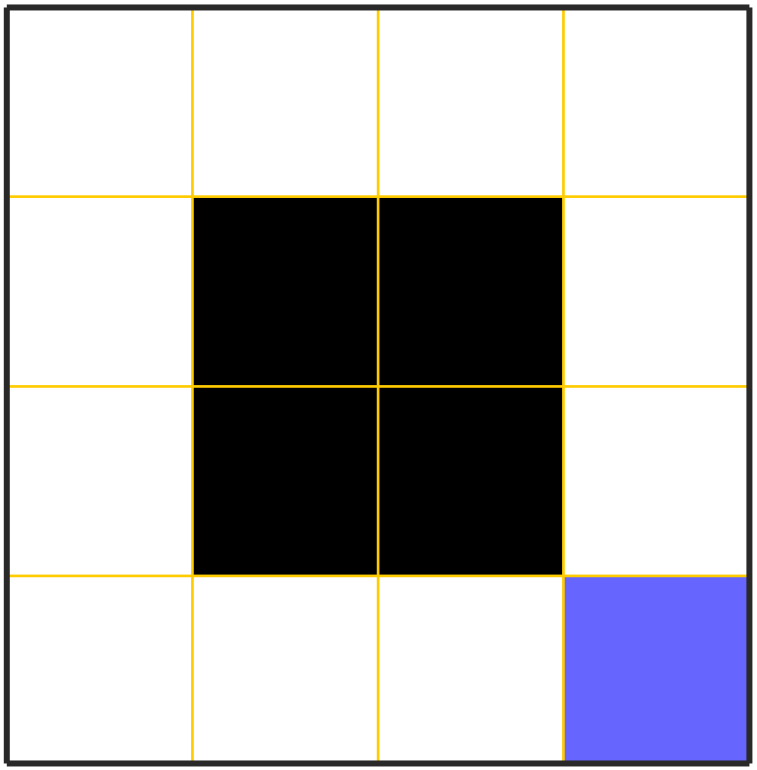
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

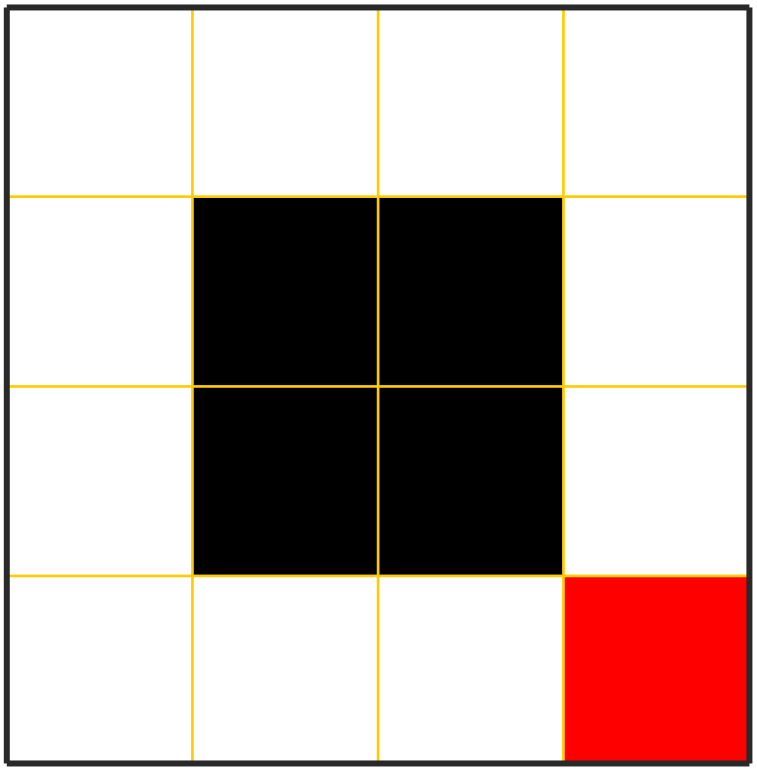
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

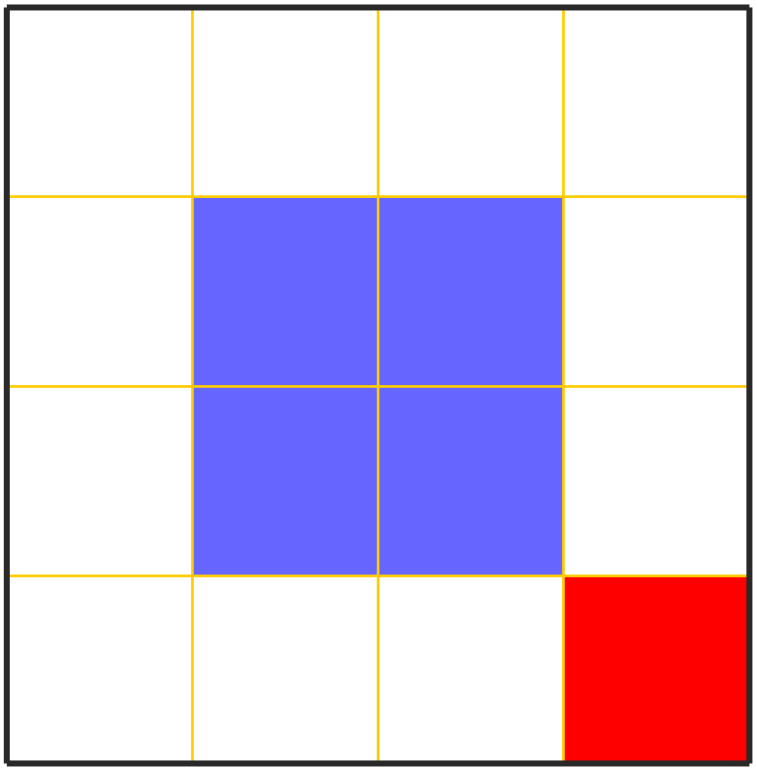
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

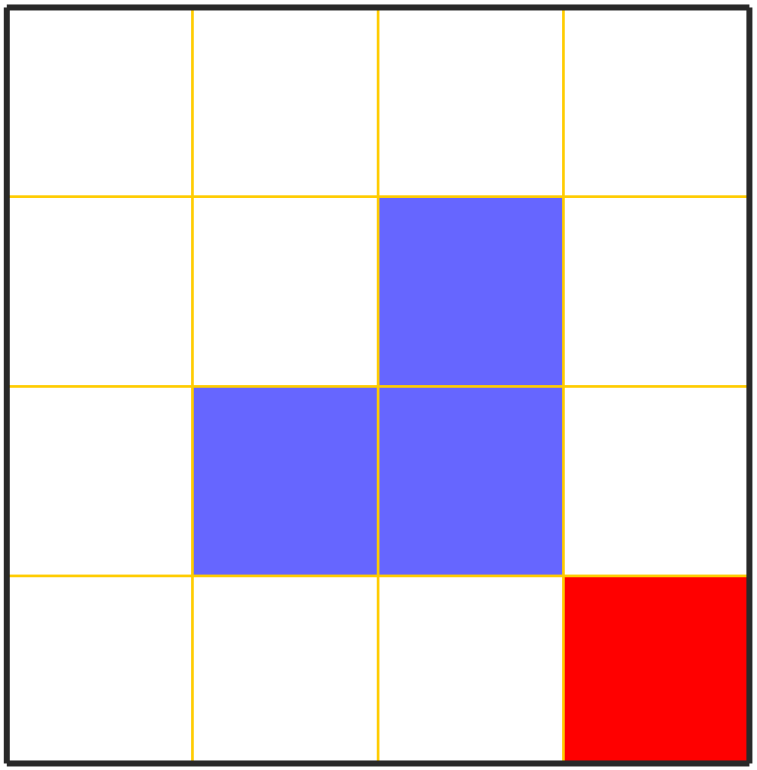
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

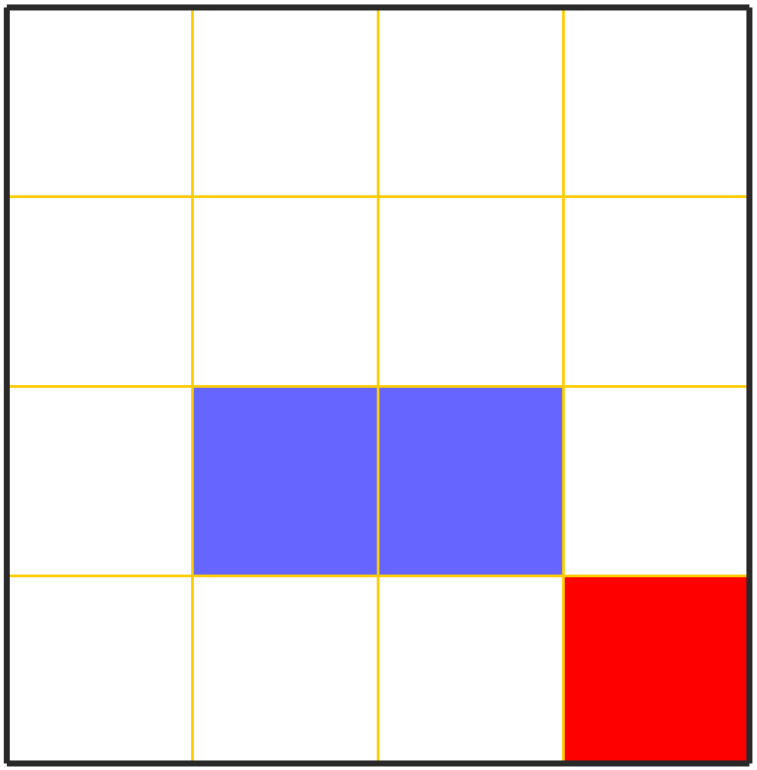
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

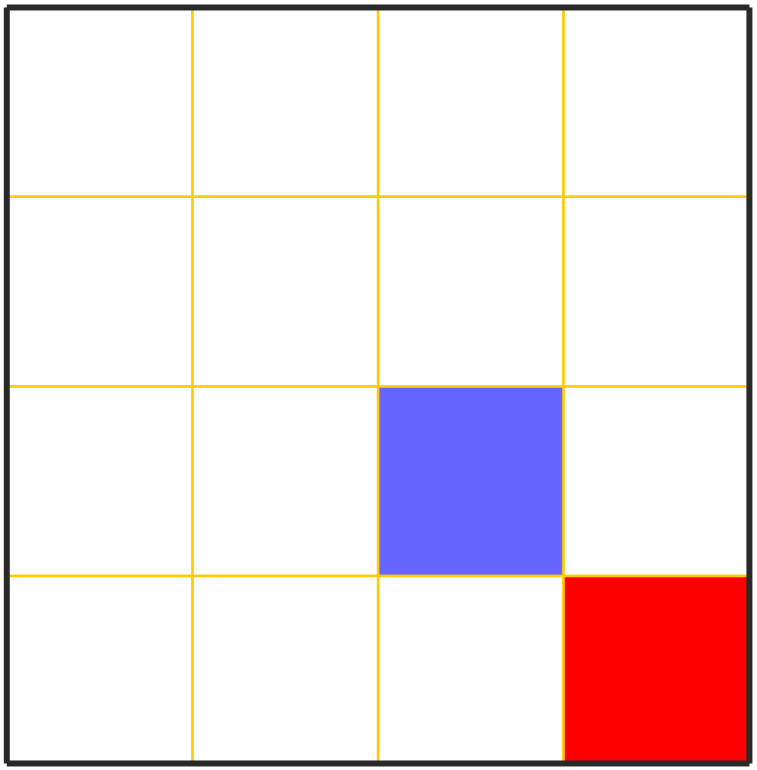
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

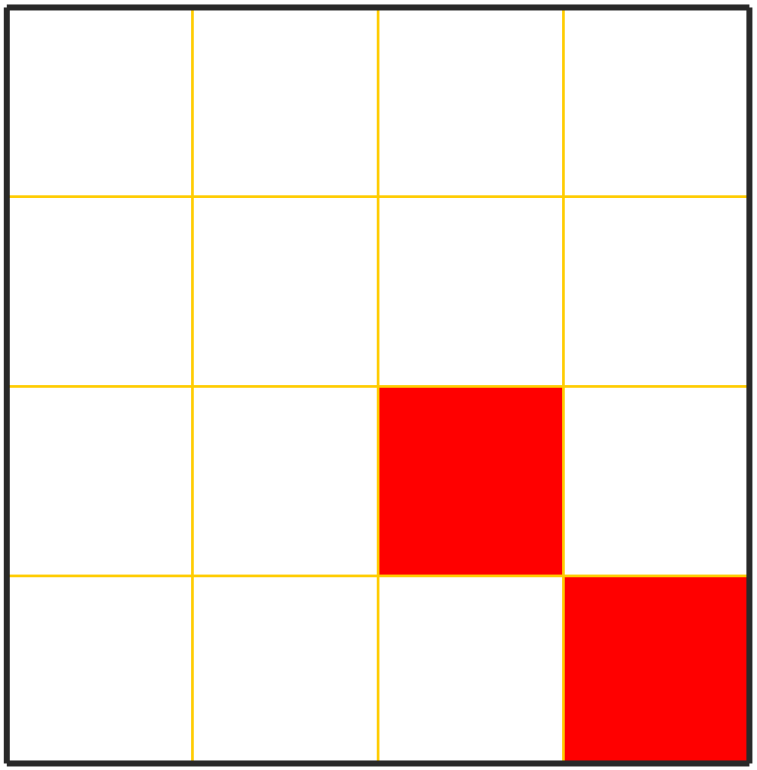
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*forward scan*

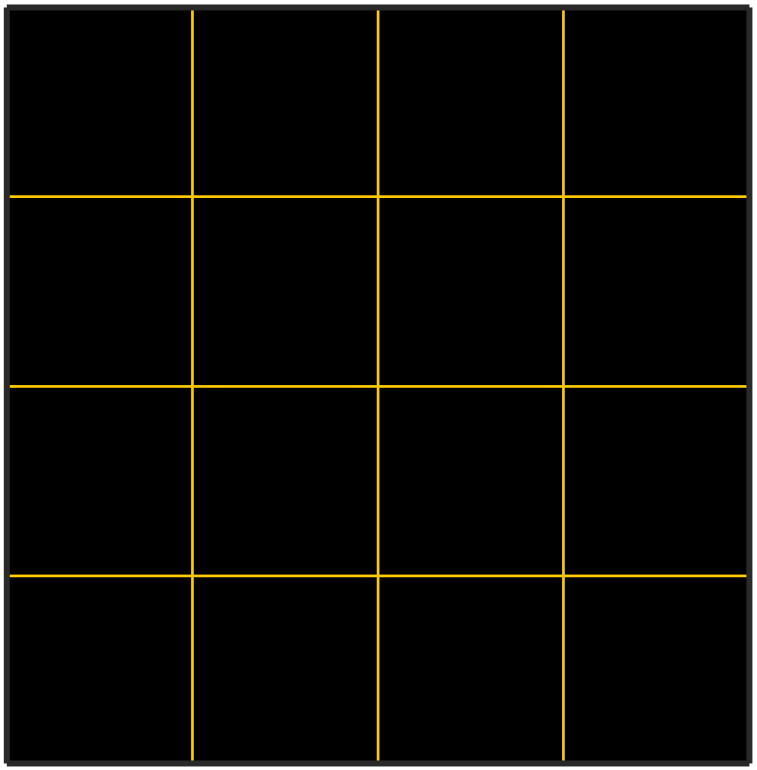
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

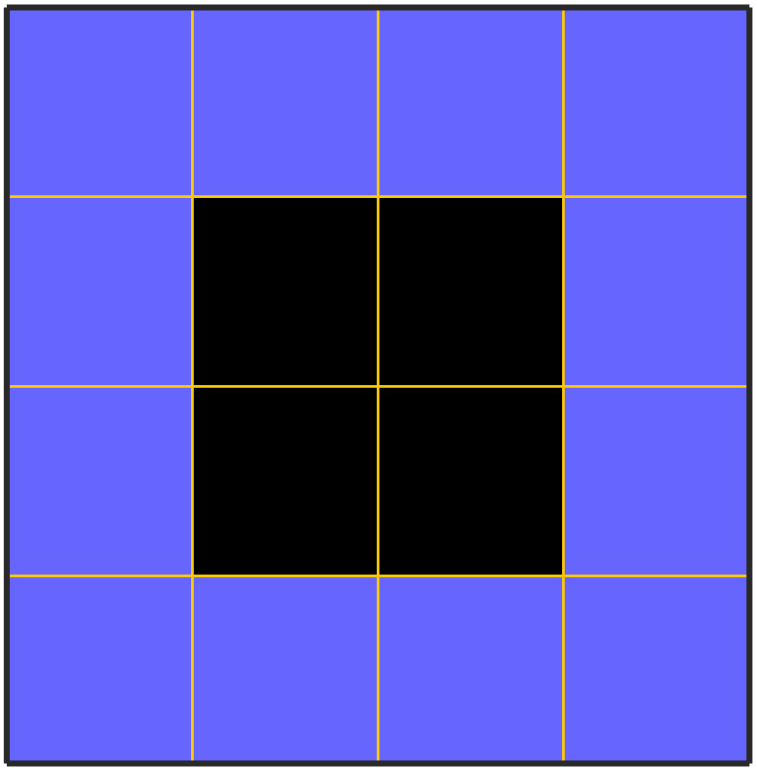
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

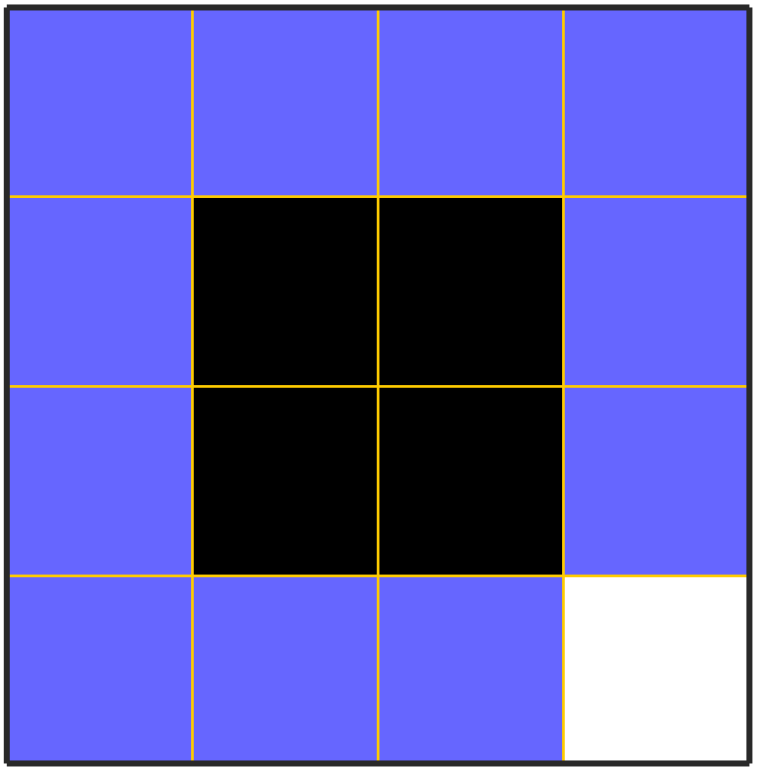
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

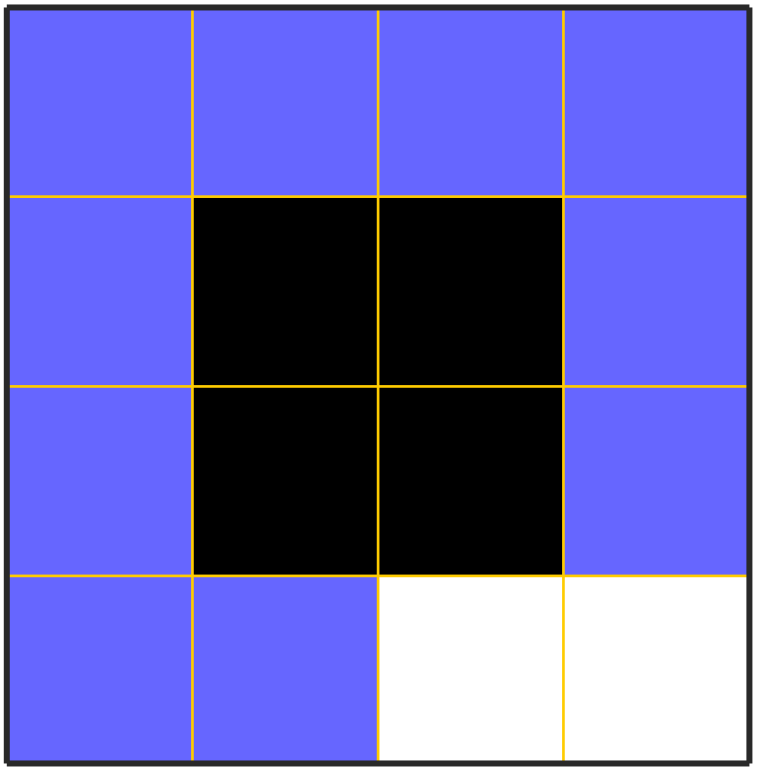
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

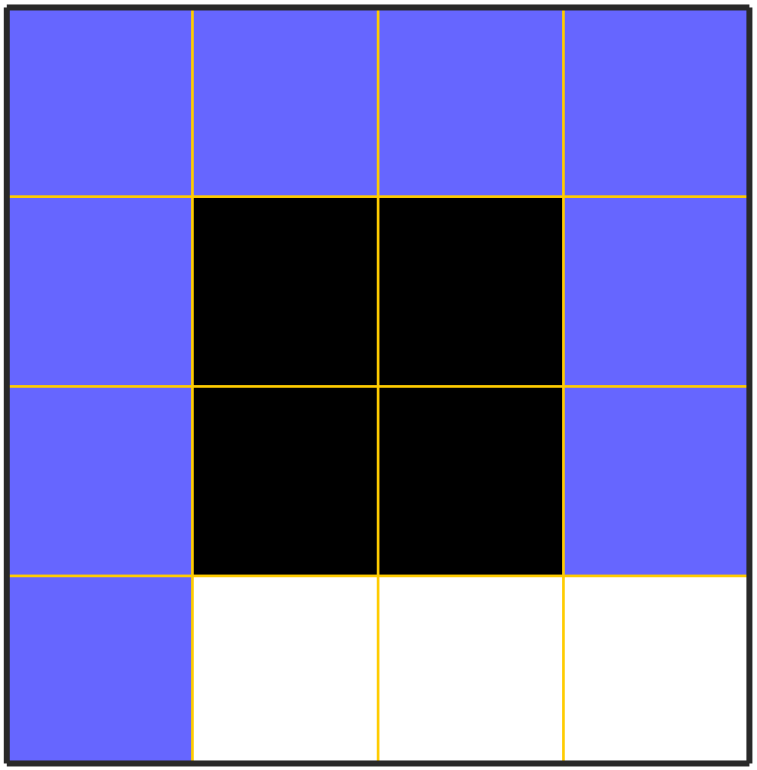
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

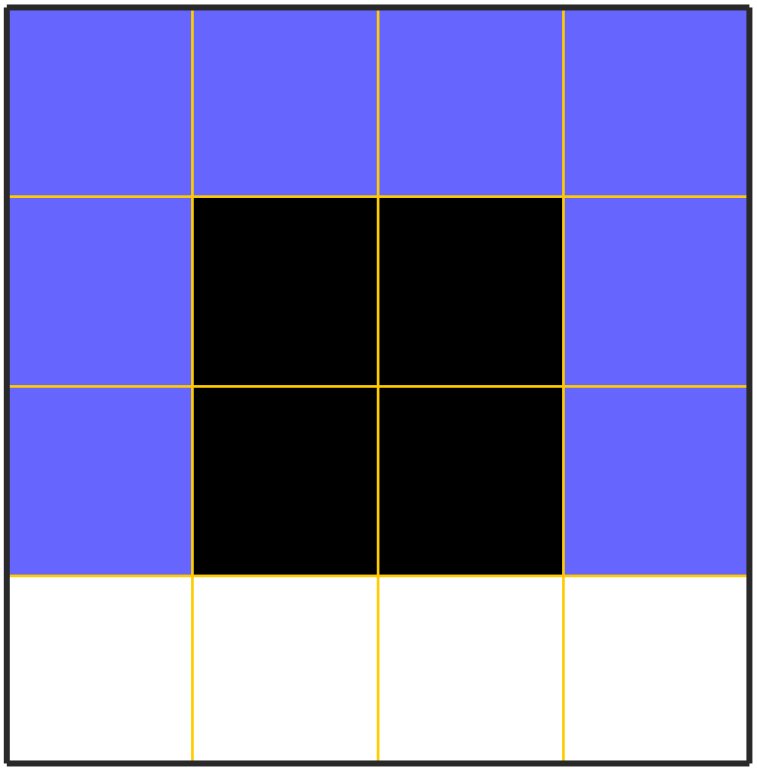
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

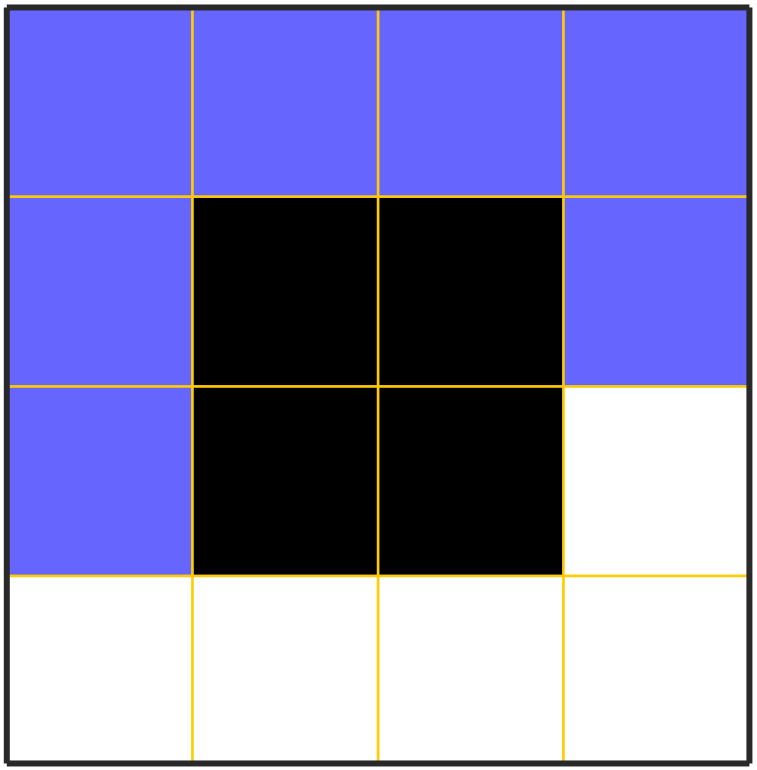
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

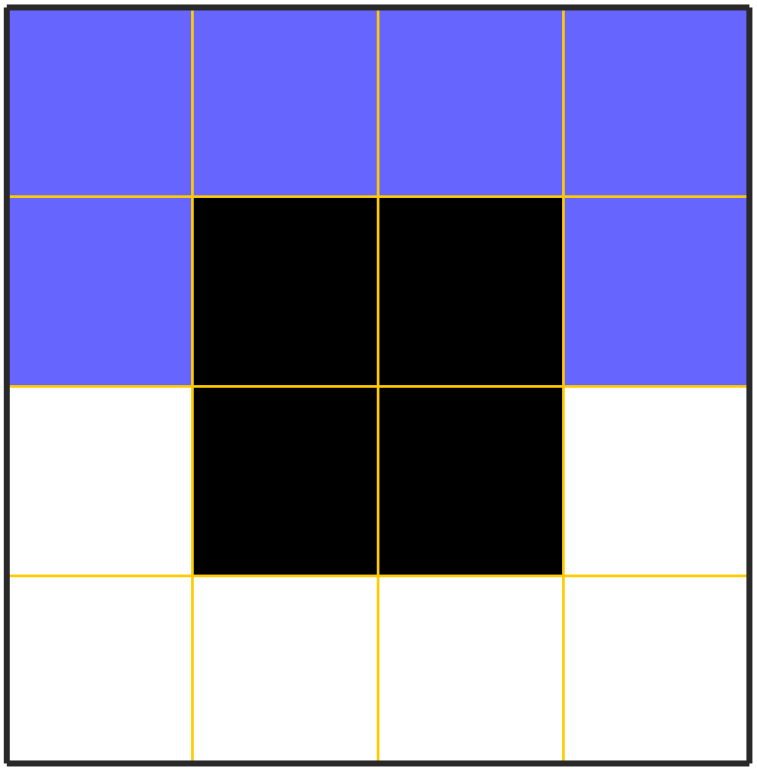
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

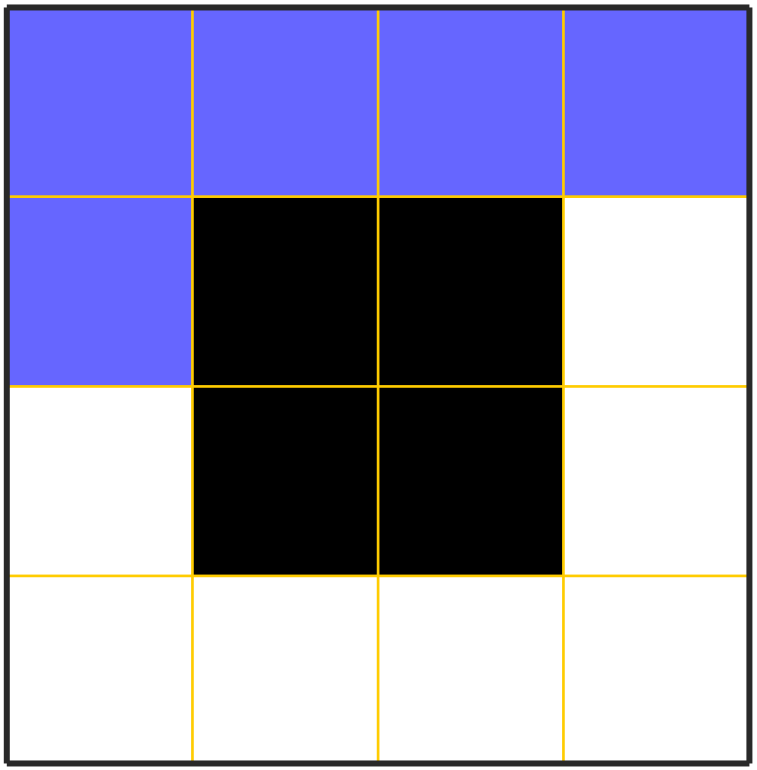
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

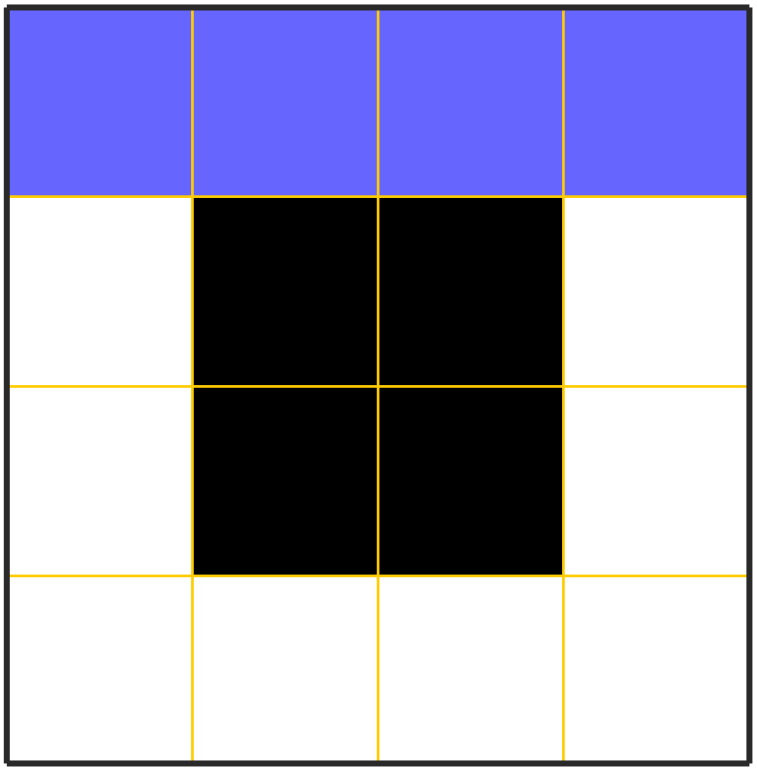
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

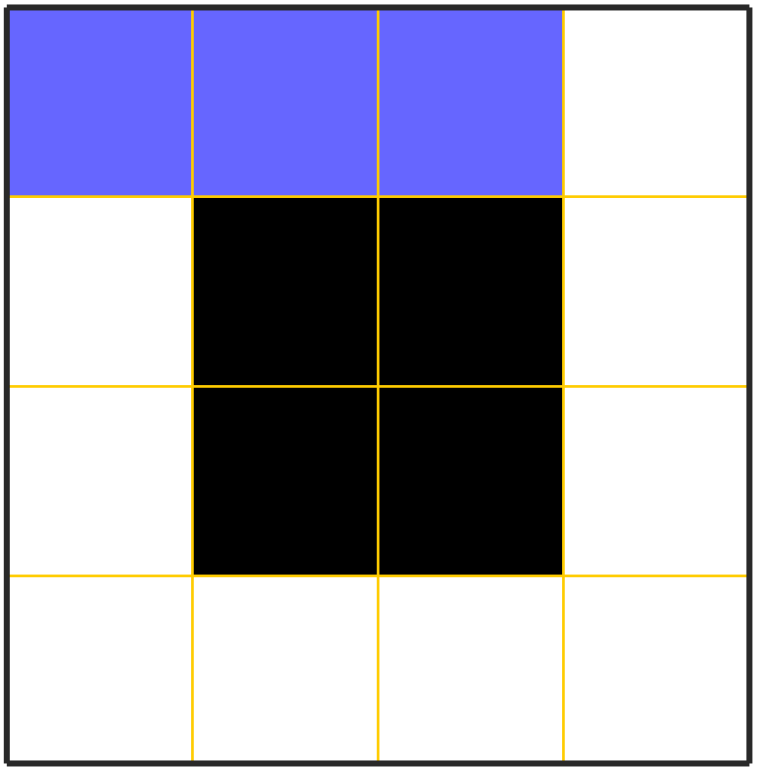
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

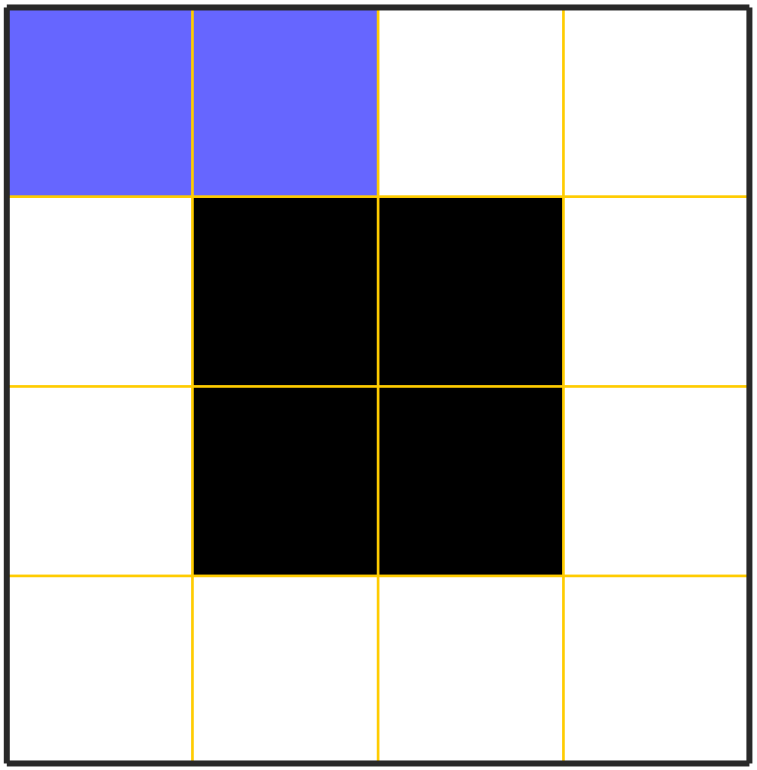
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

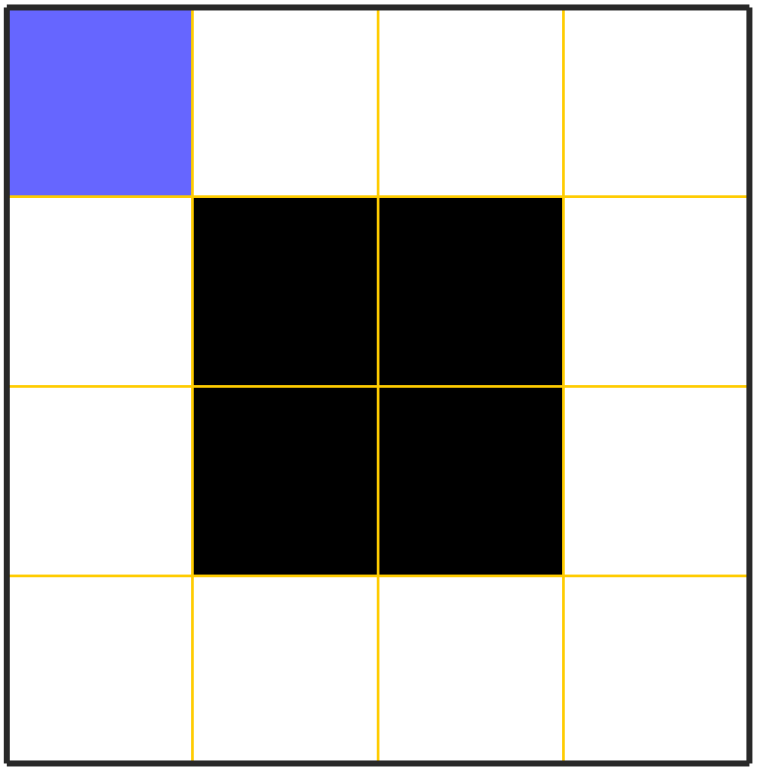
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

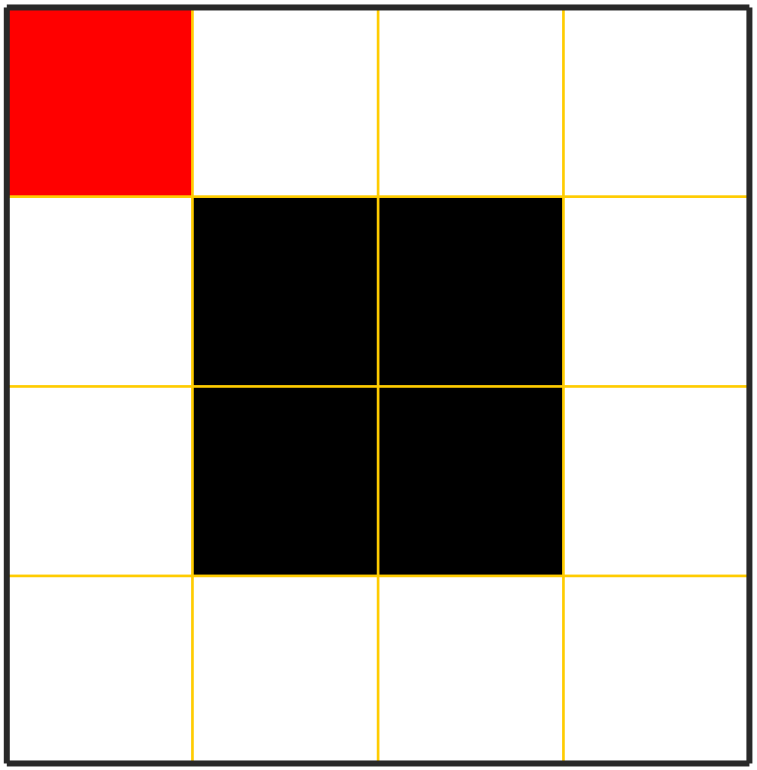
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

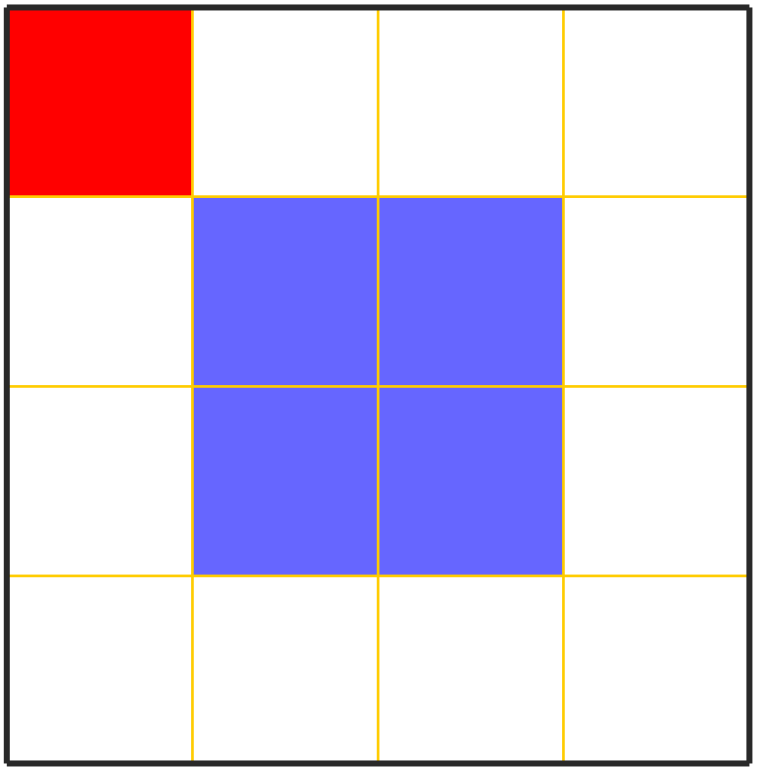
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

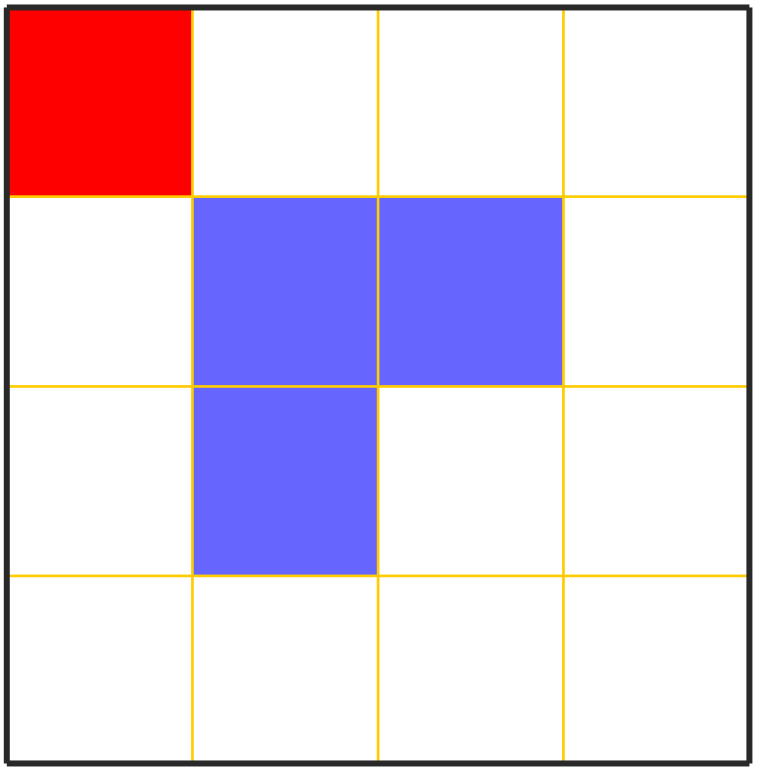
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

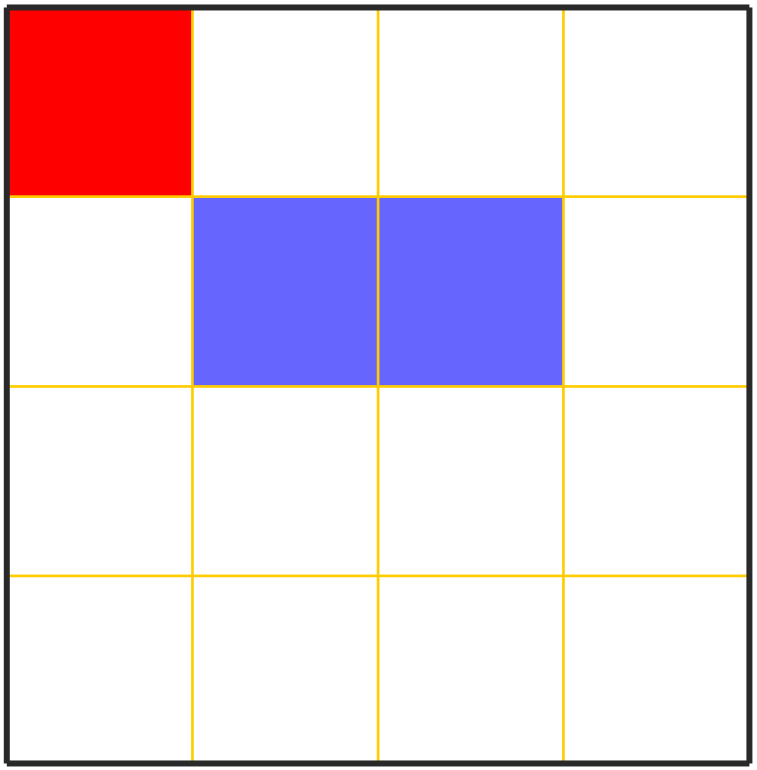
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

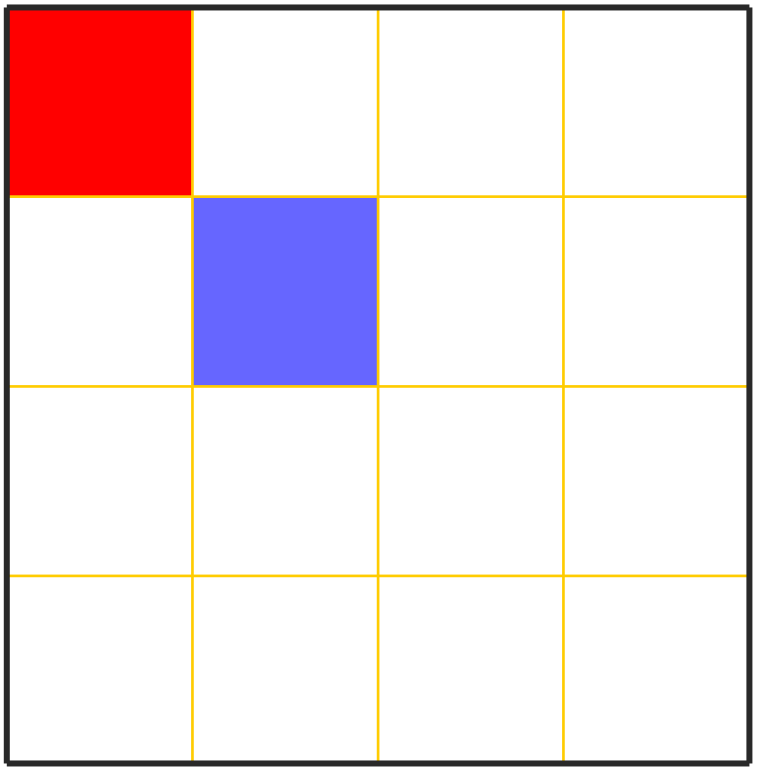
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

*backward scan*

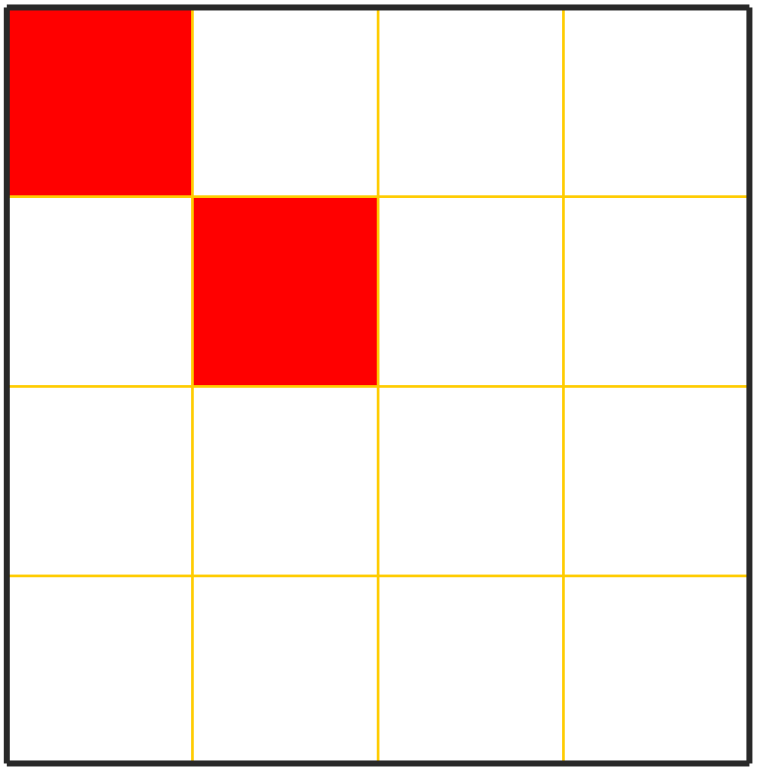
**border points**
**inner points**
**skeletal points**



# Problem of order-independency

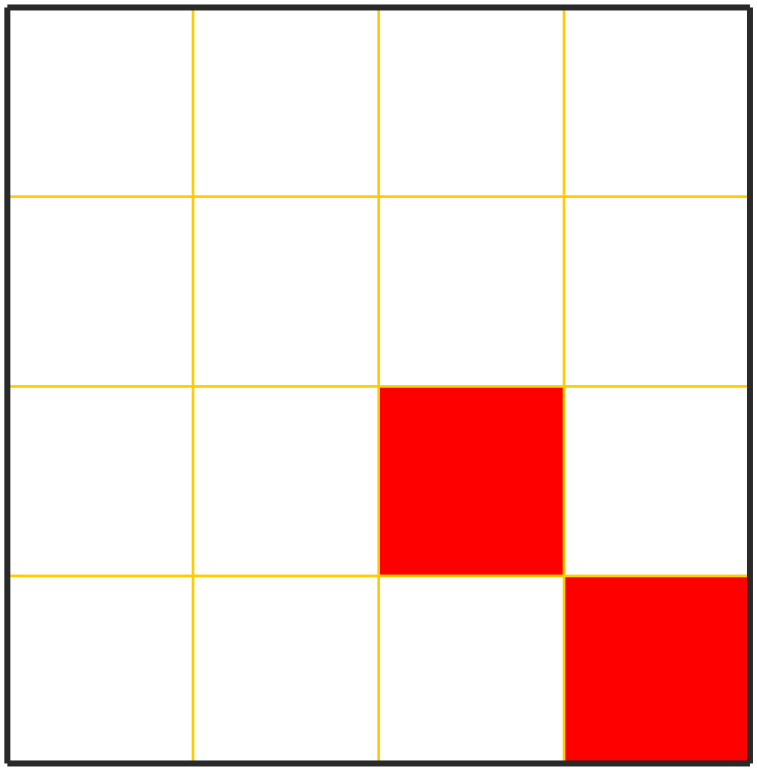
*backward scan*

**border points**
**inner points**
**skeletal points**

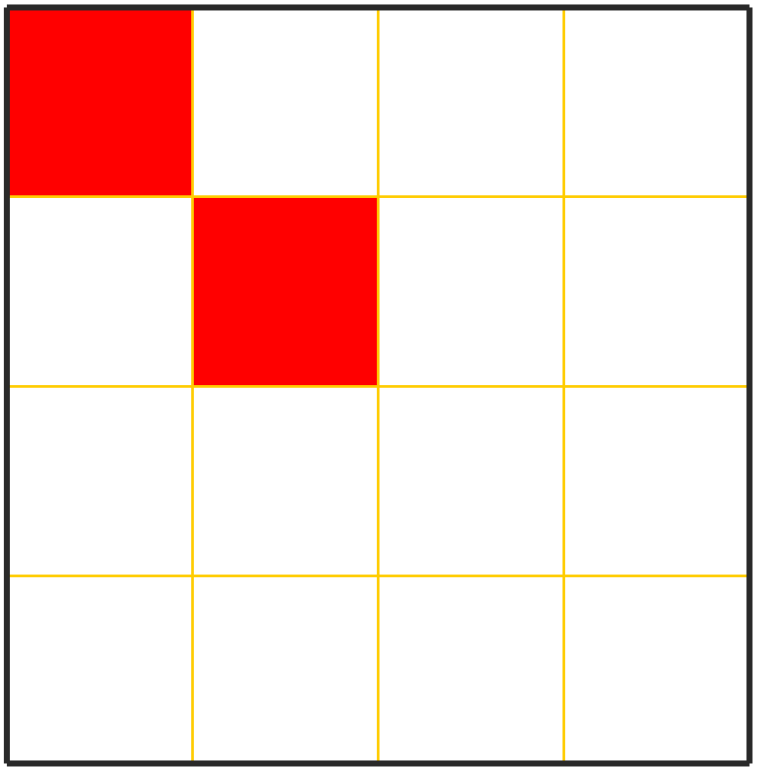


# Problem of order-independency

*forward scan*

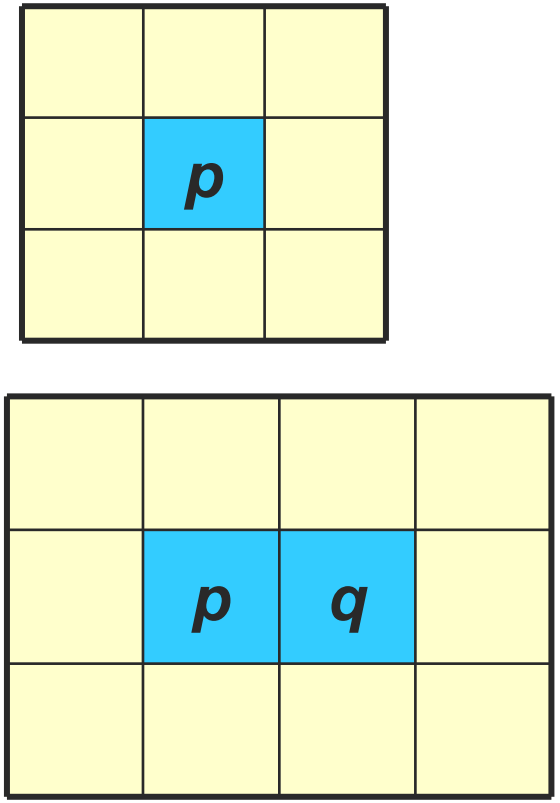


*backward scan*

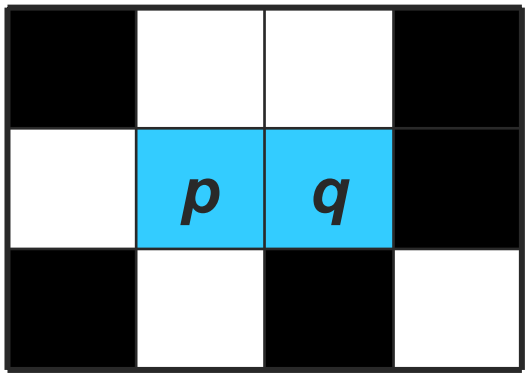


# Basic notions

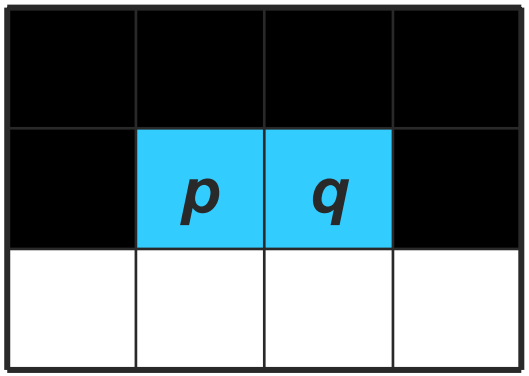
- ***C(p)*: number of black components in the 3x3 environment of *p***
- ***C(p,q)*: number of black components in the 3x4 or 4x3 environment of the set {*p*,*q*}**



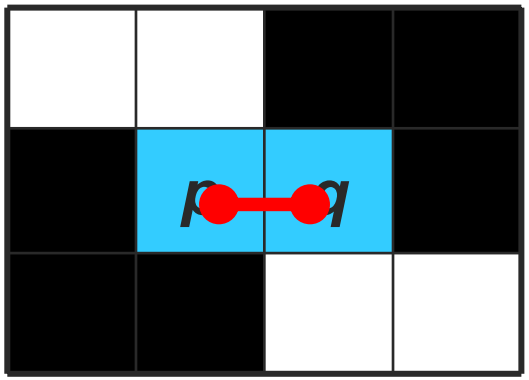
# Some examples




C(p)=3
C(q)=1
C(p,q)=3




C(p)=1
C(q)=1
C(p,q)=1




C(p)=1
C(q)=1
C(p,q)=2

*critical pair*

# Expected properties

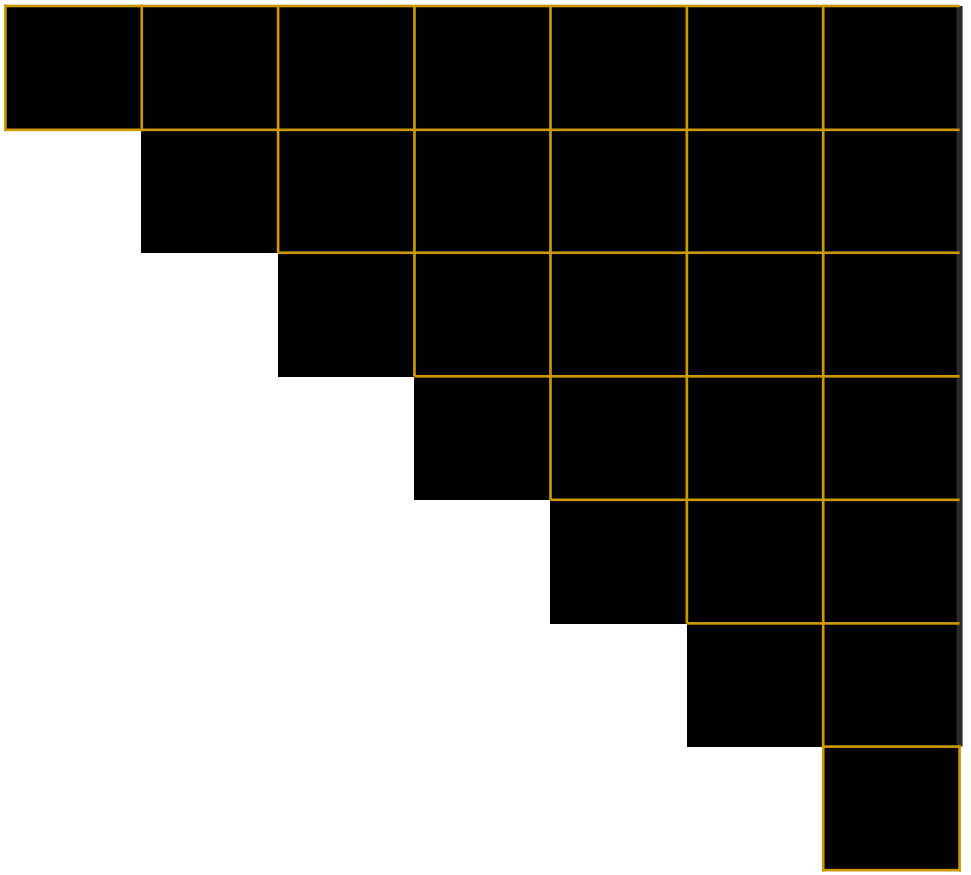
- Order-independency
- Topology preserving
- 1-pixel thin
- Shape preserving

# Order-independent algorithms

- 2D
  - Algorithm of Ranwez and Soille (RS)
    - Basically a *shrinking* algorithm!
    - End-point detection needed (anchor points)
  - Our proposed algorithm
    - End-point definition included
- 3D
  - Unsolved problem

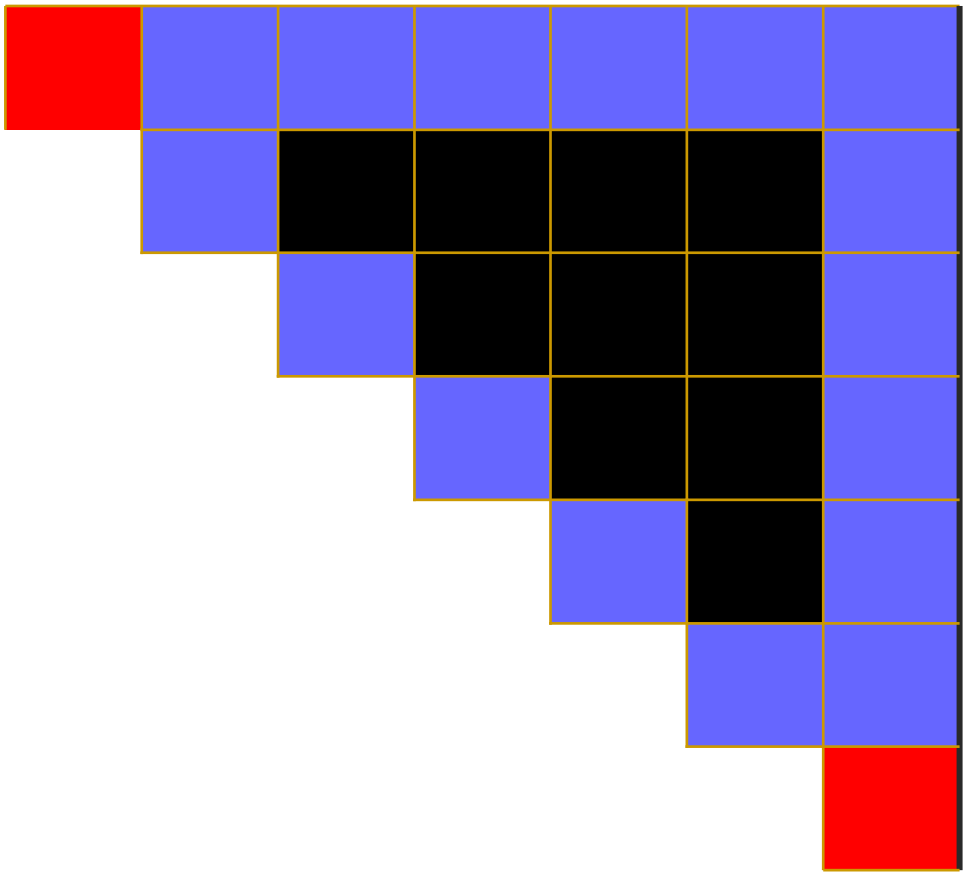
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



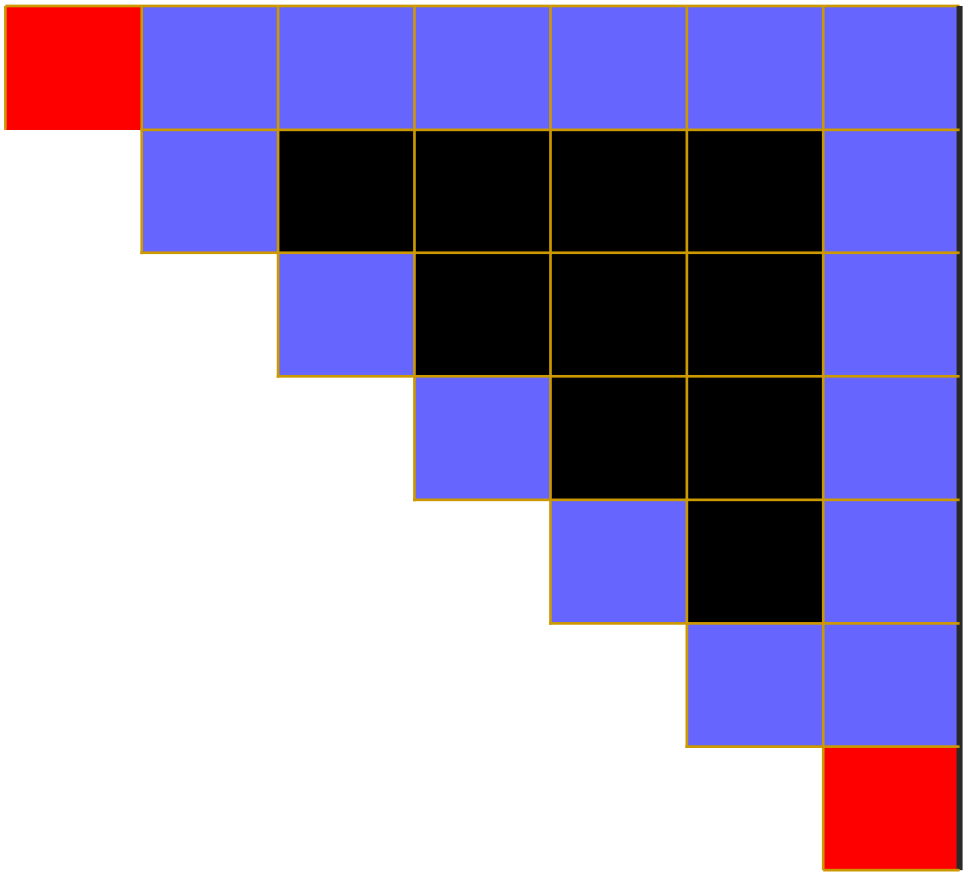
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



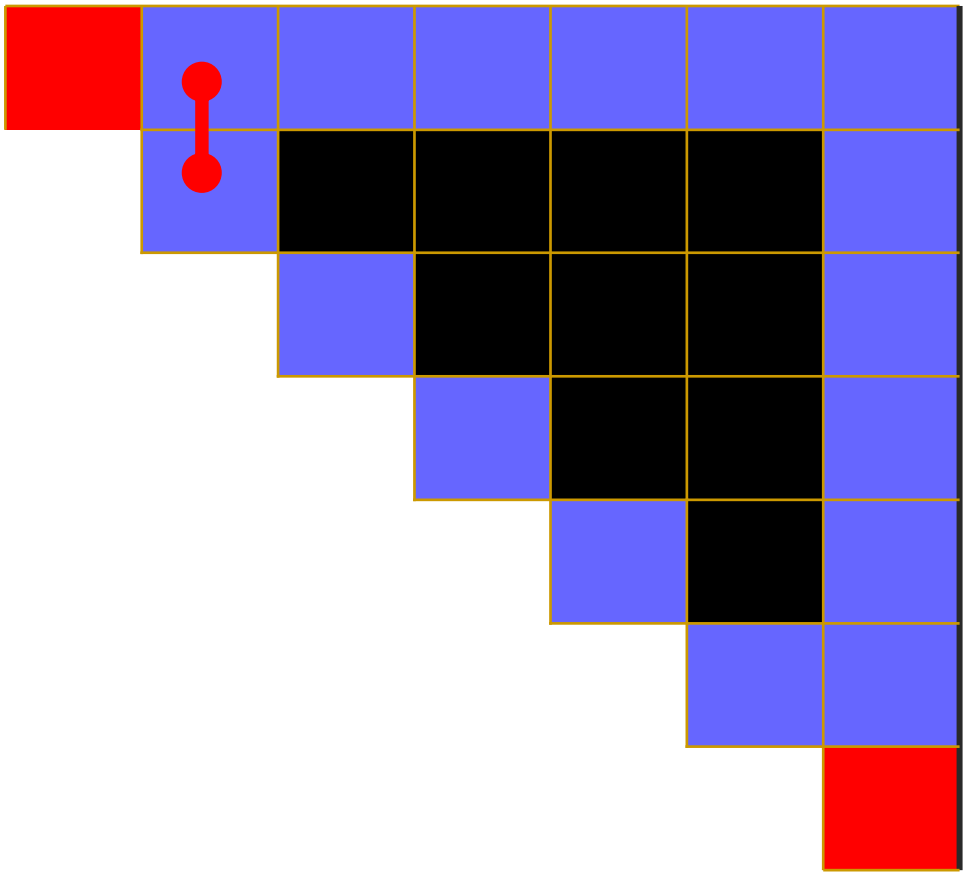
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



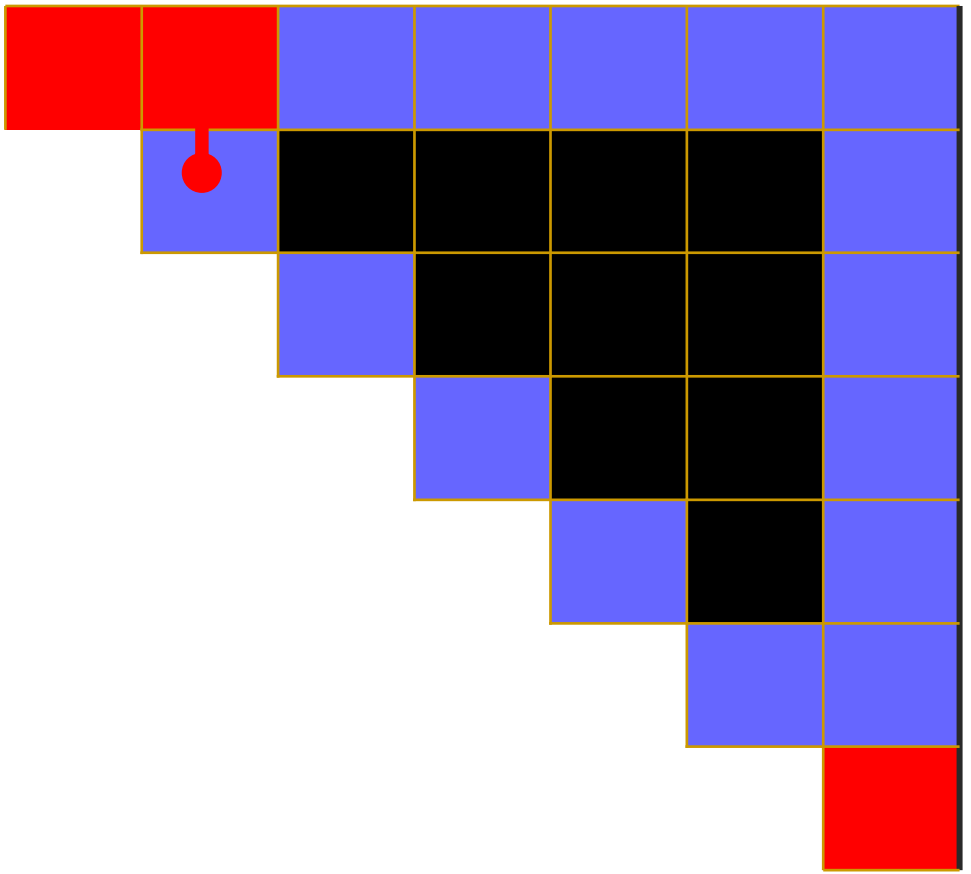
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



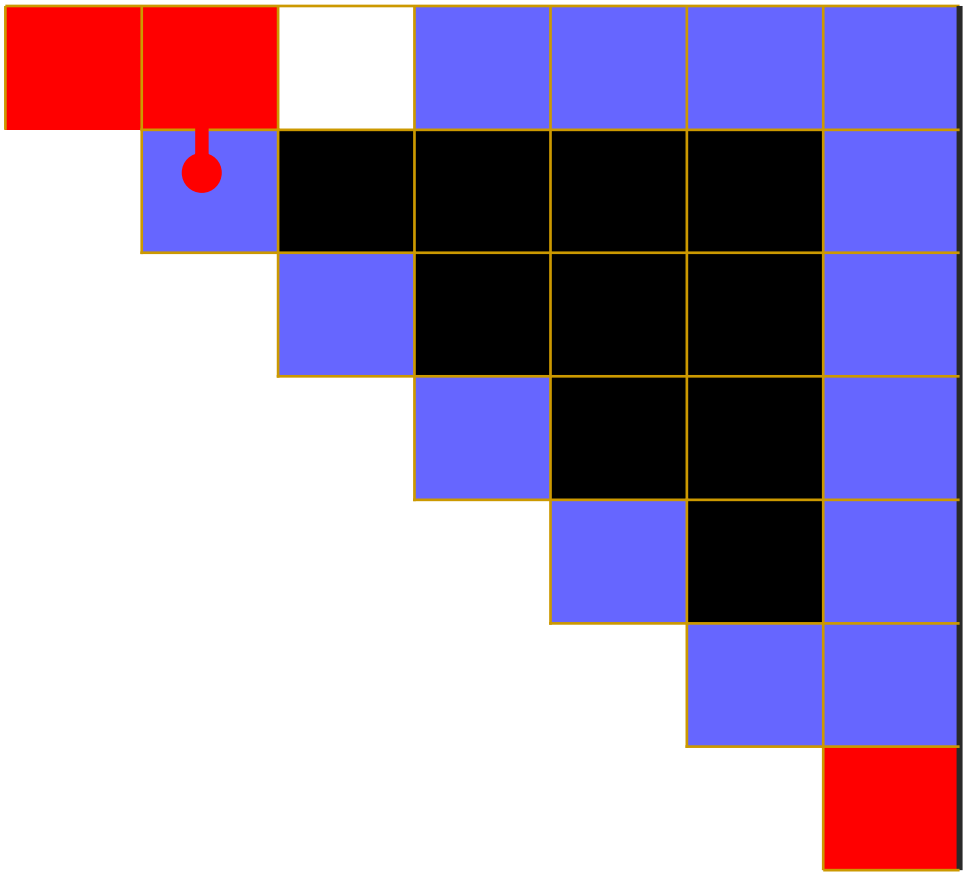
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



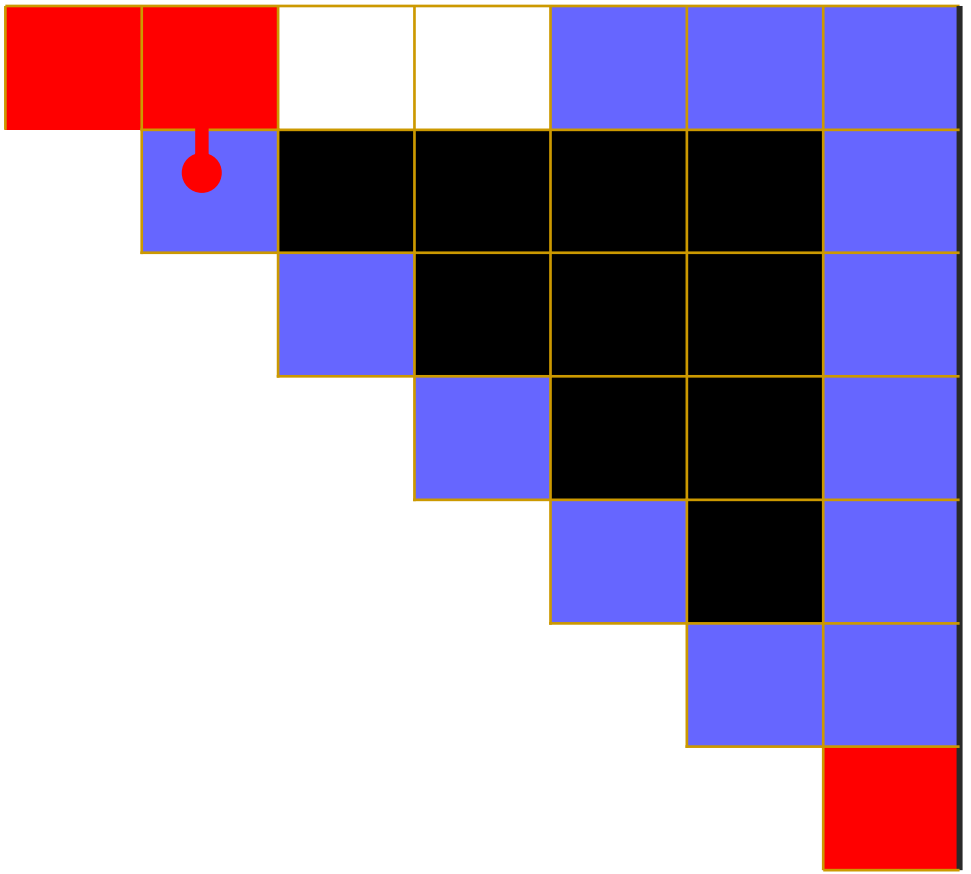
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



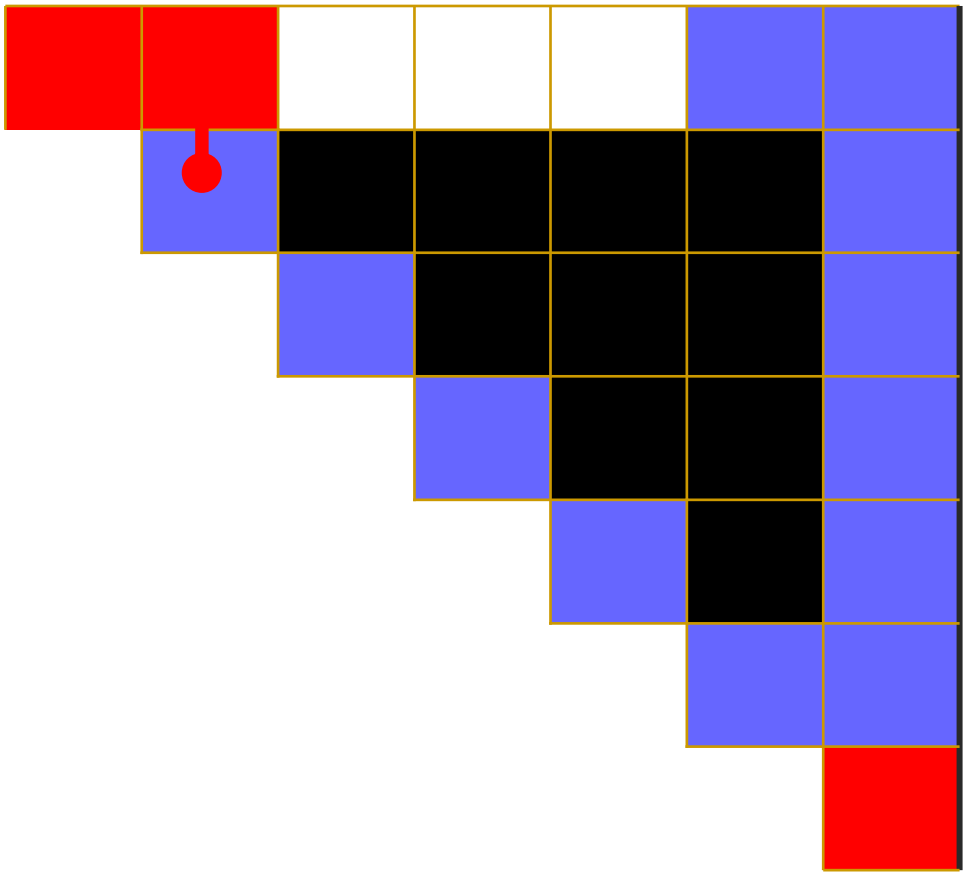
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



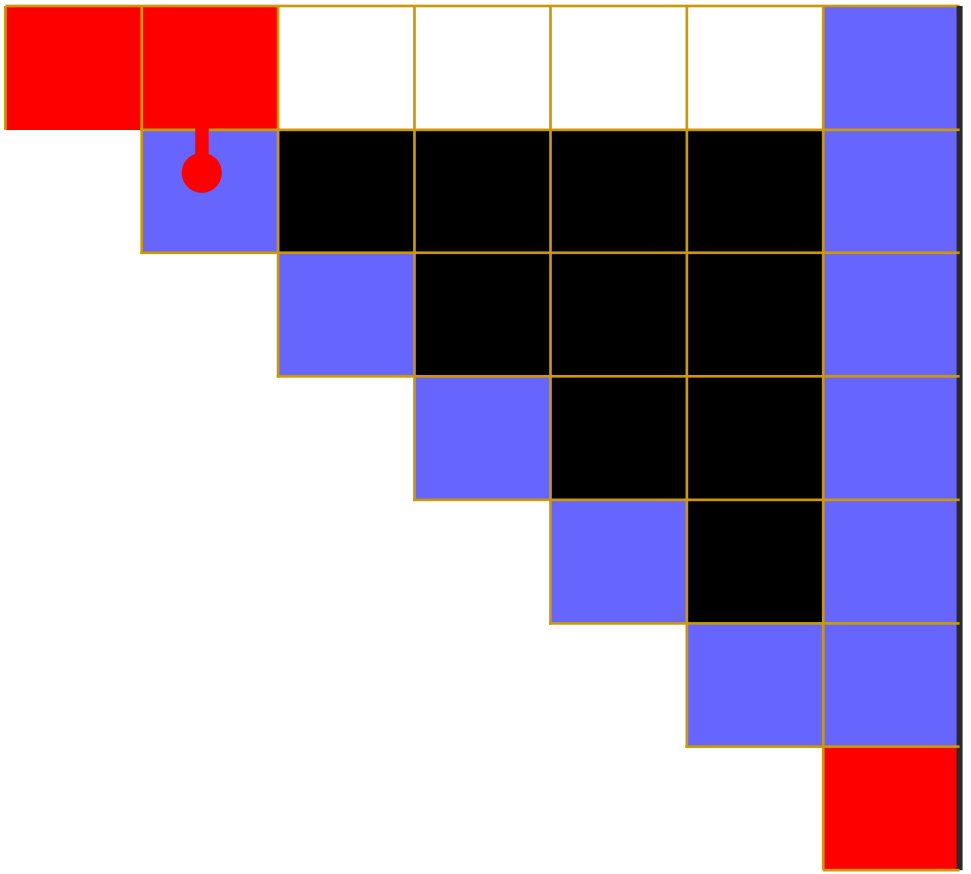
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



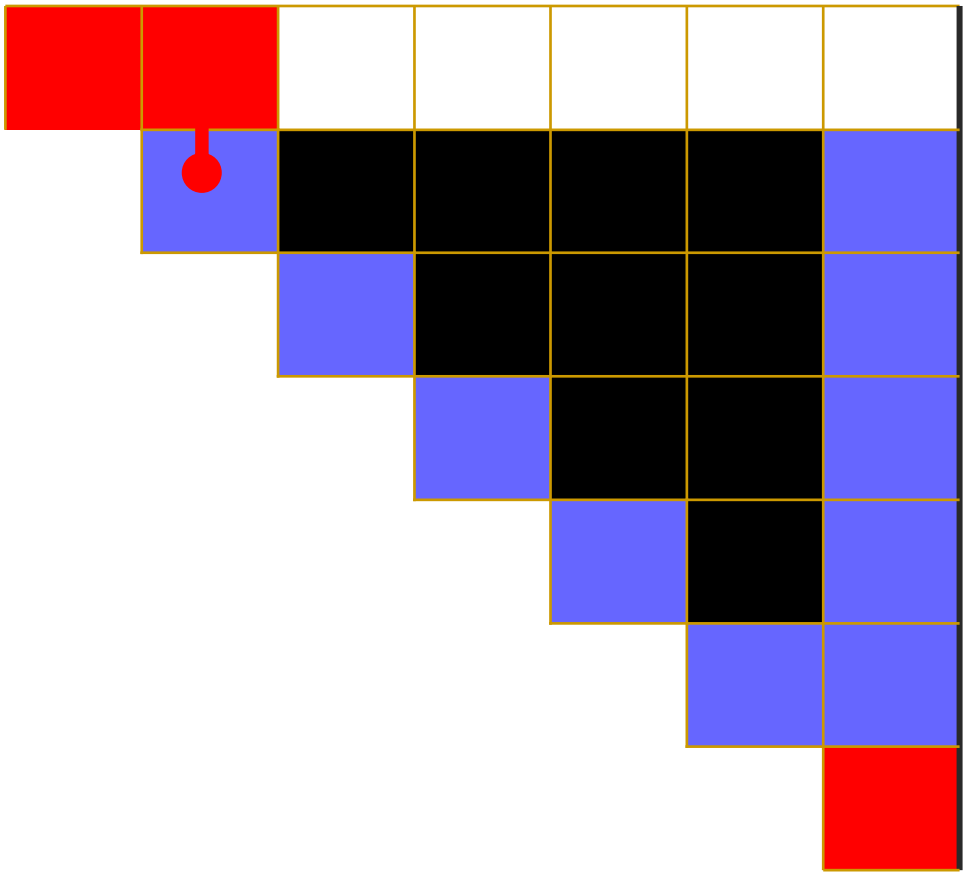
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



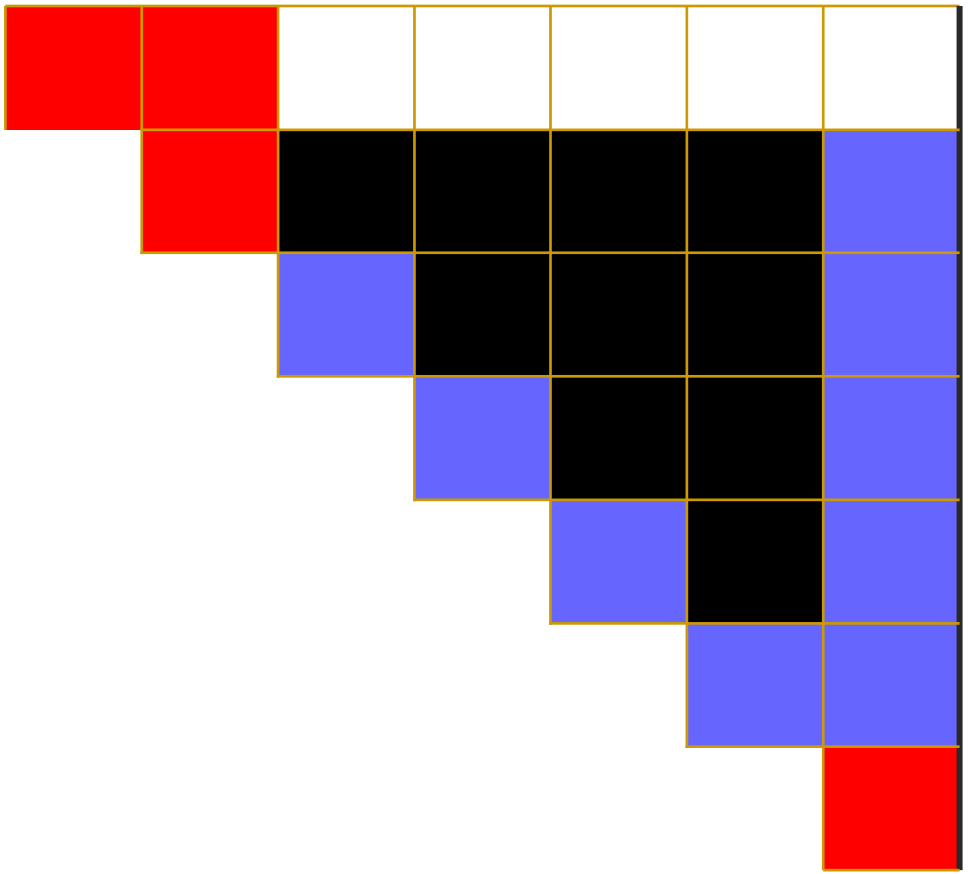
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



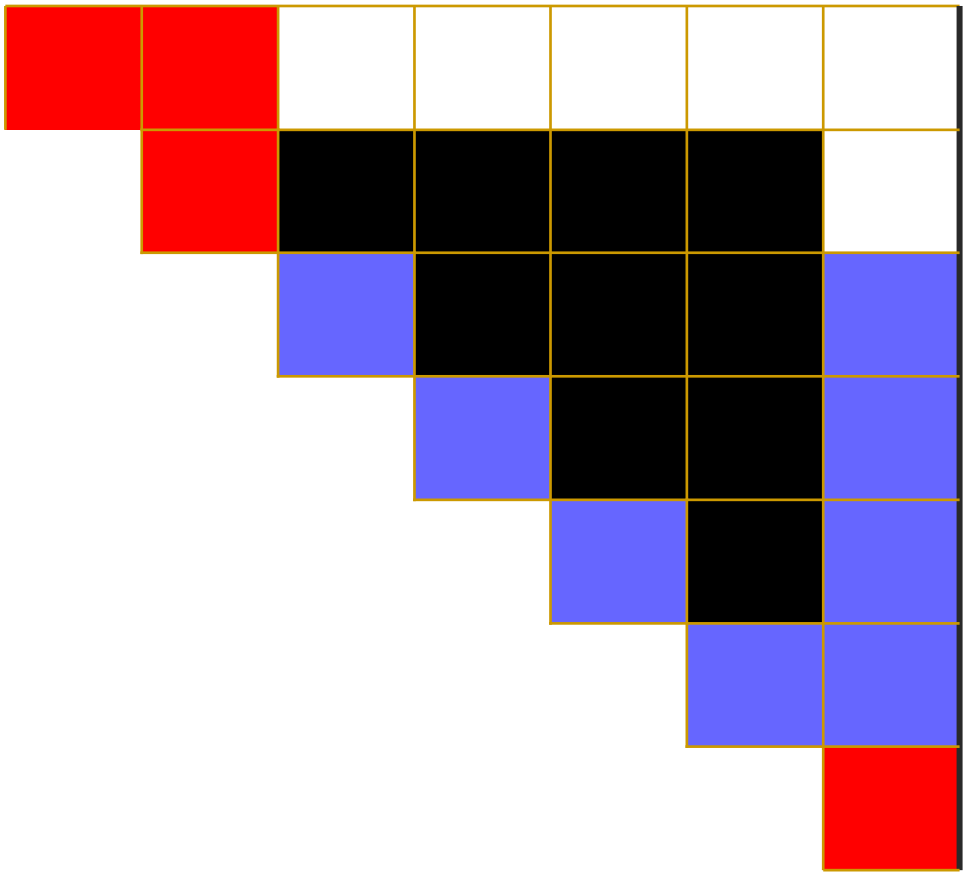
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



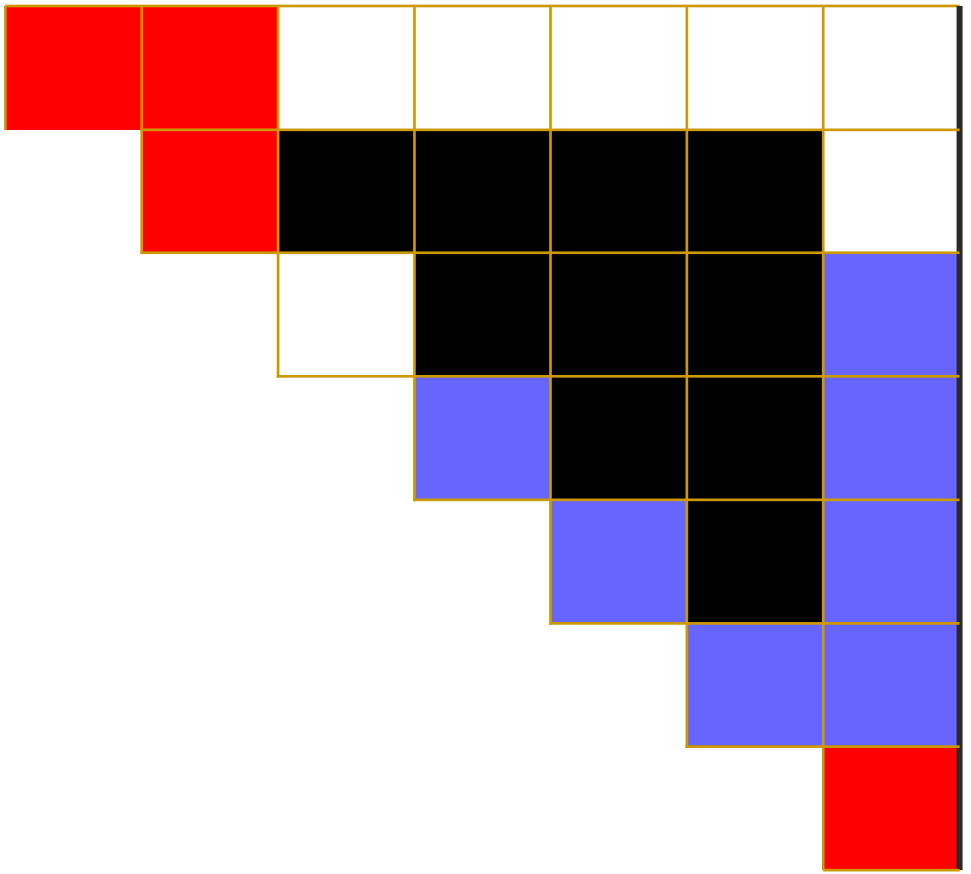
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



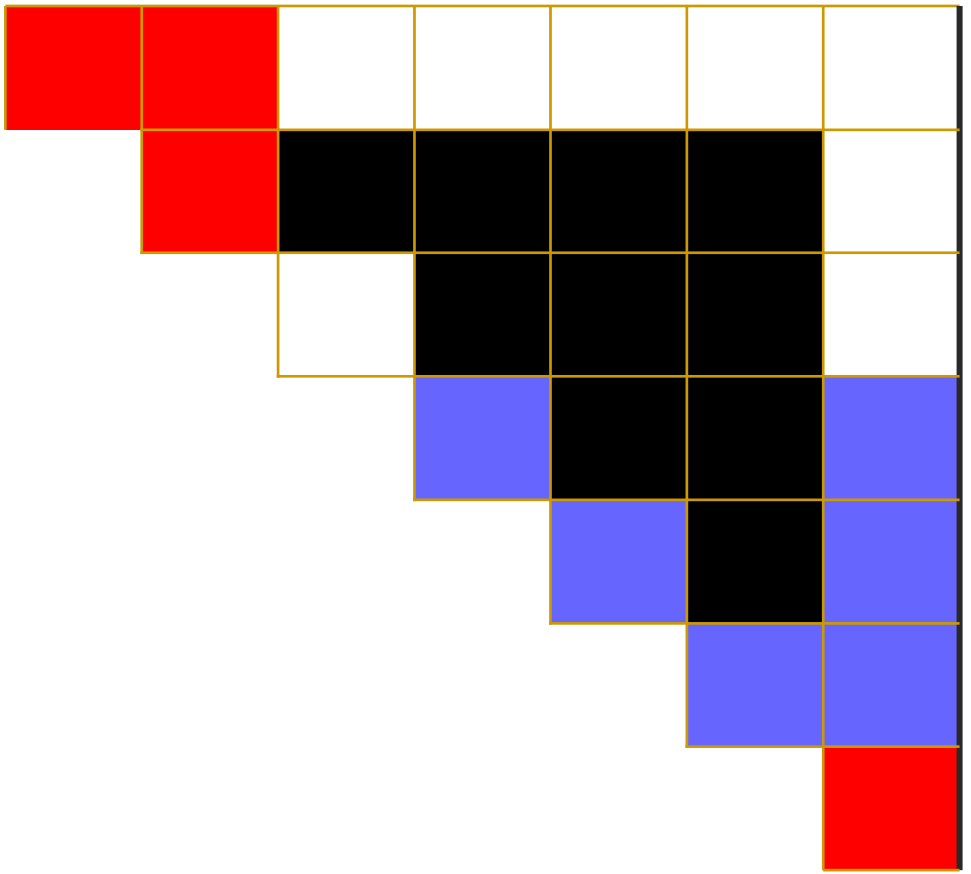
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



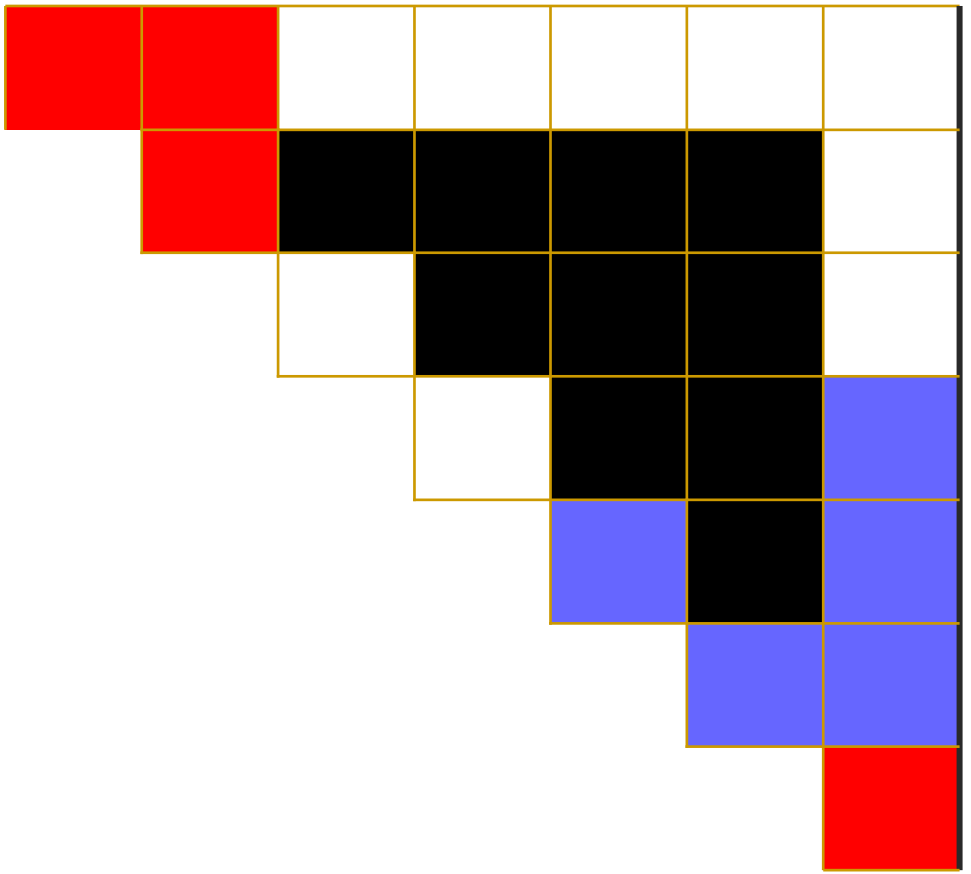
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



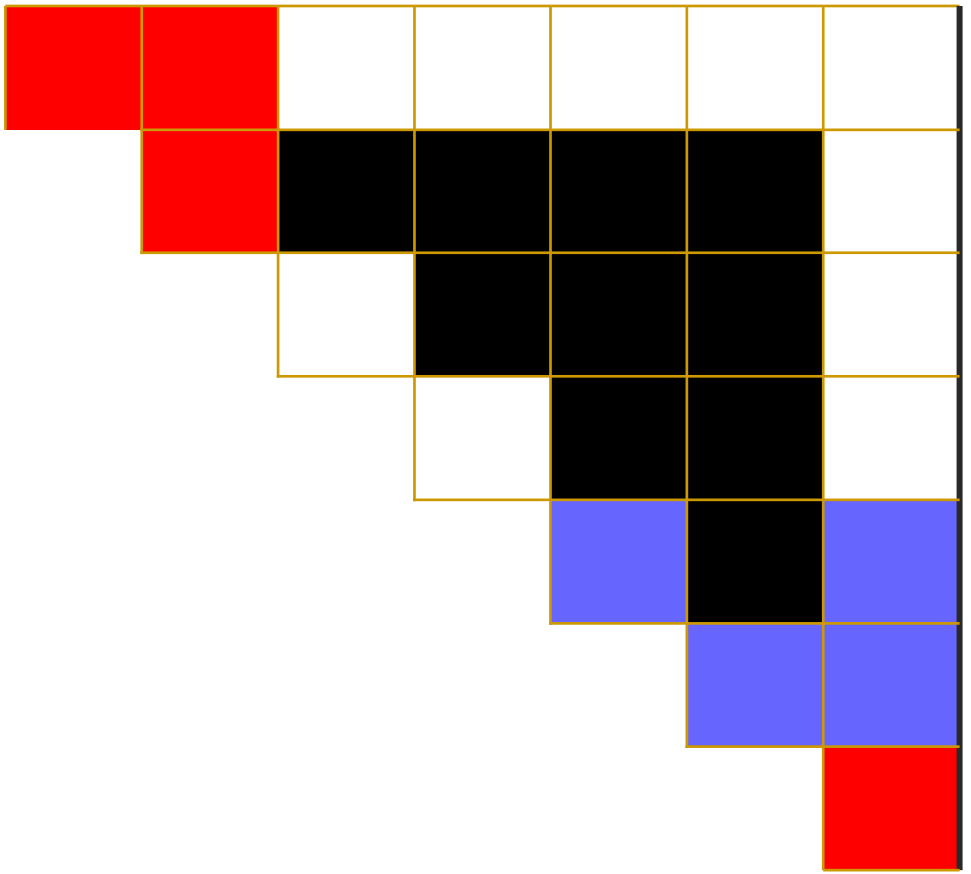
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



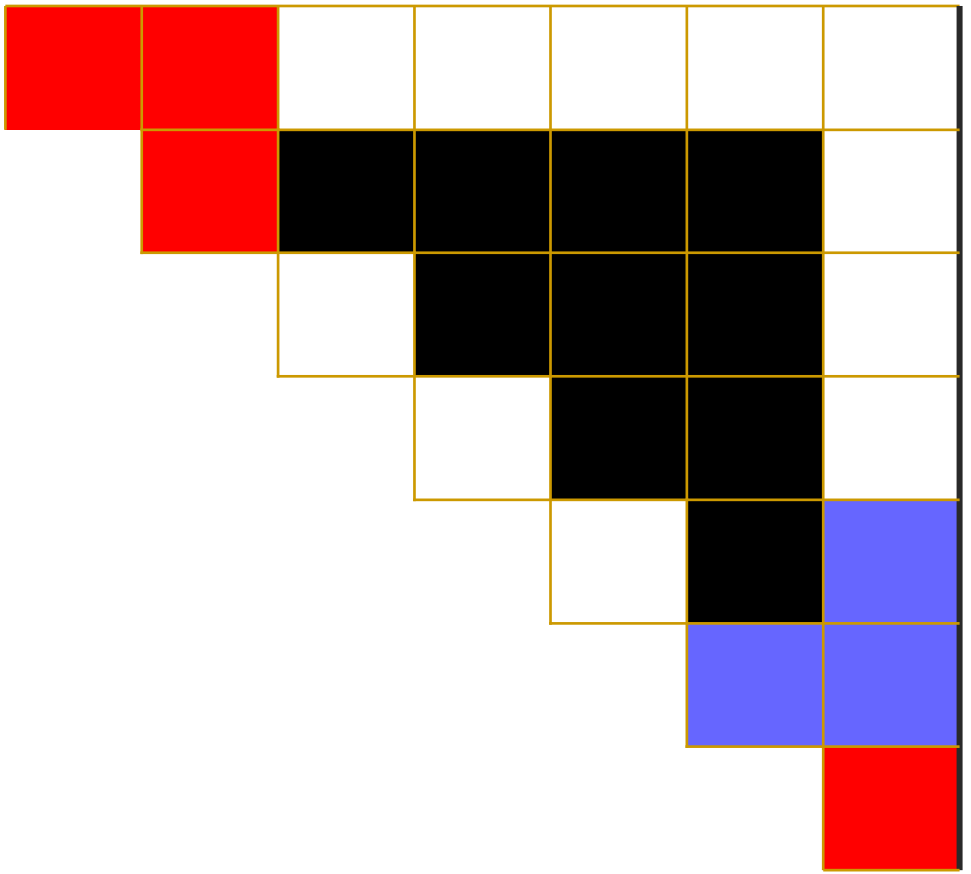
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



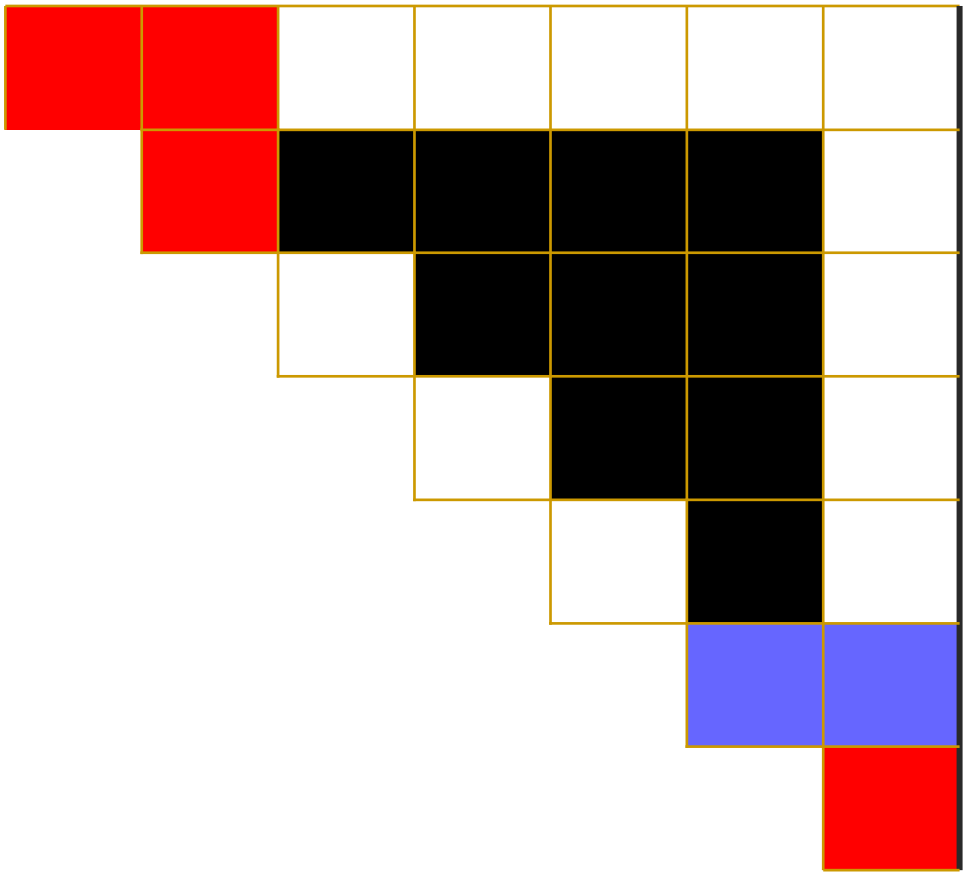
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



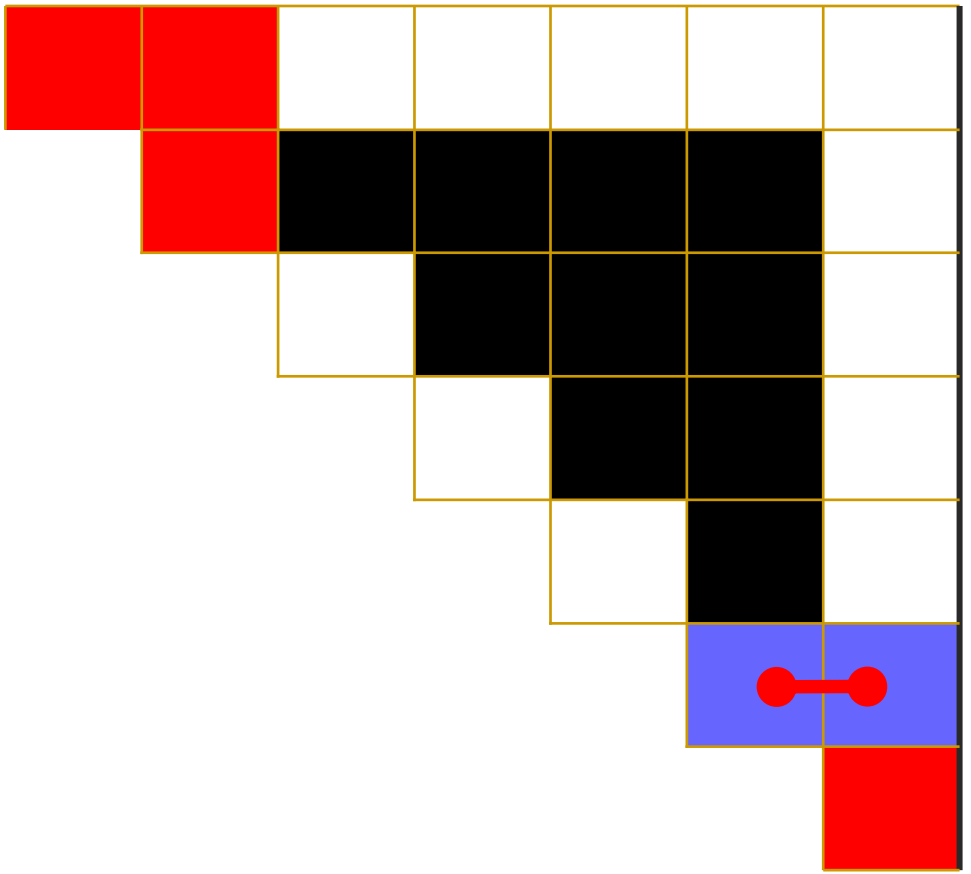
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



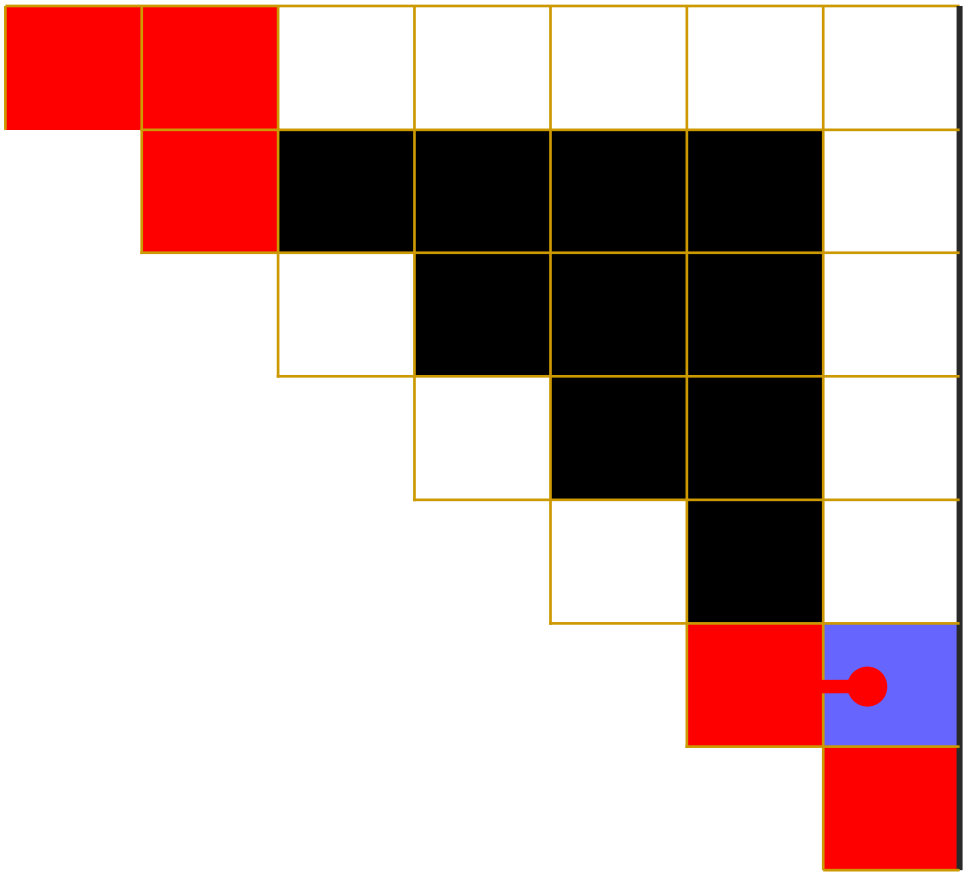
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



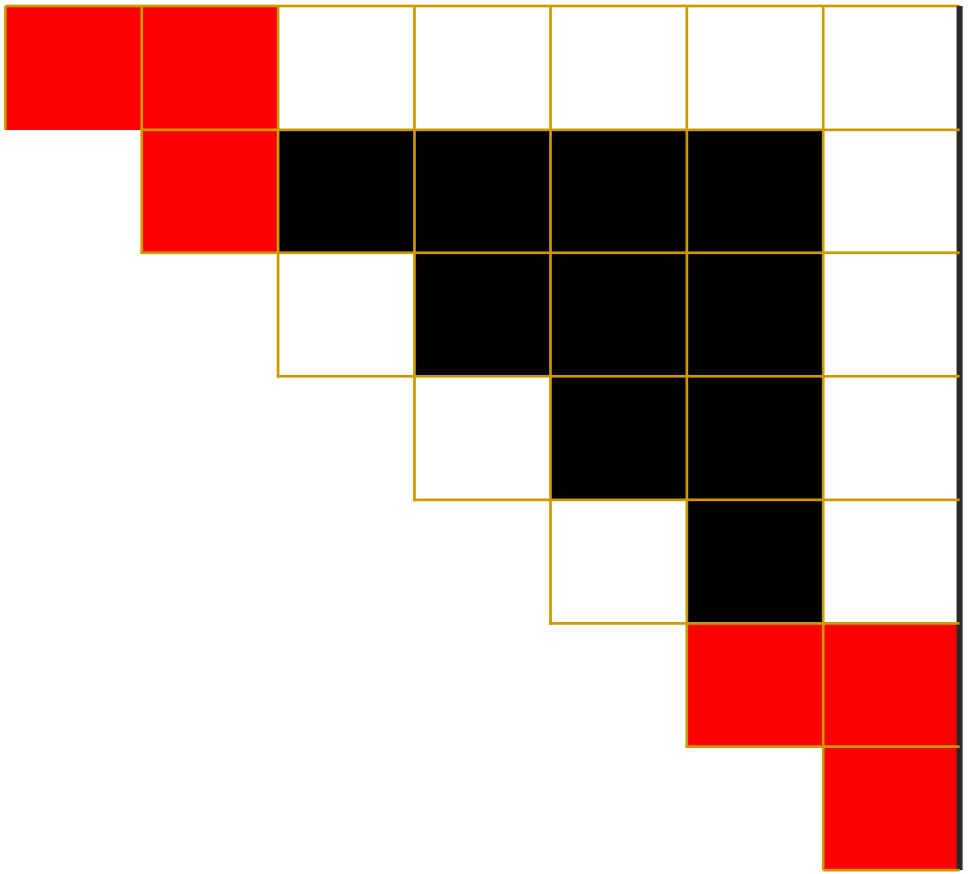
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



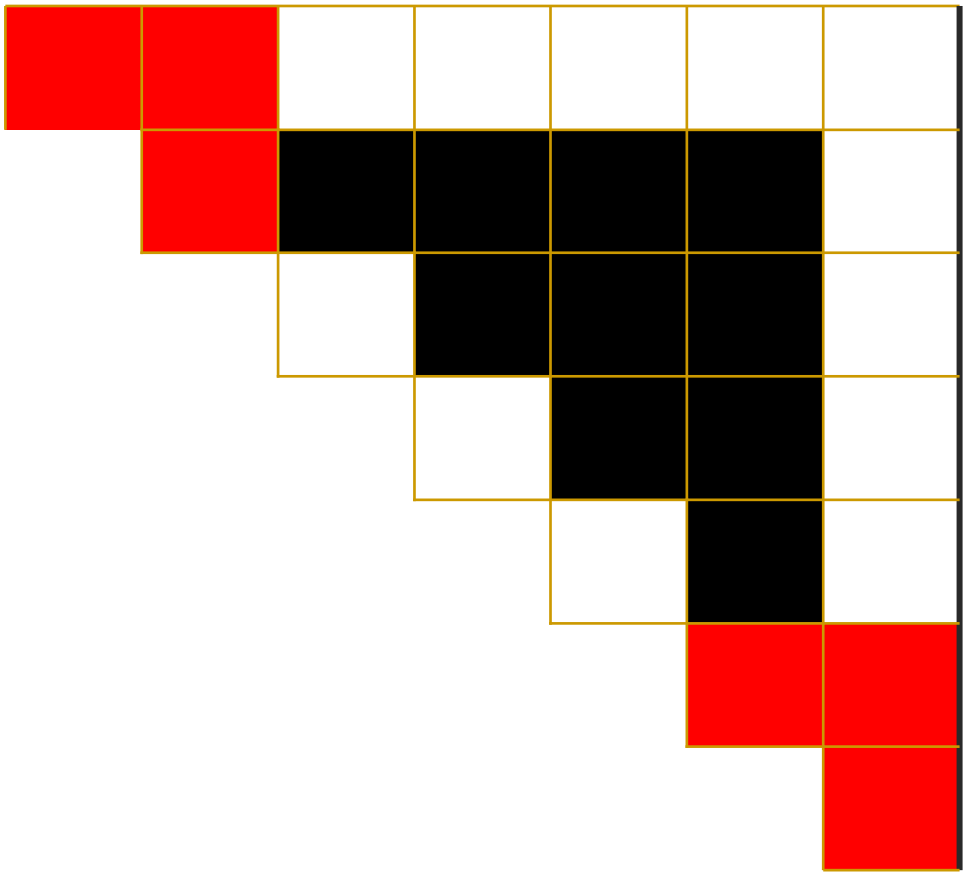
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



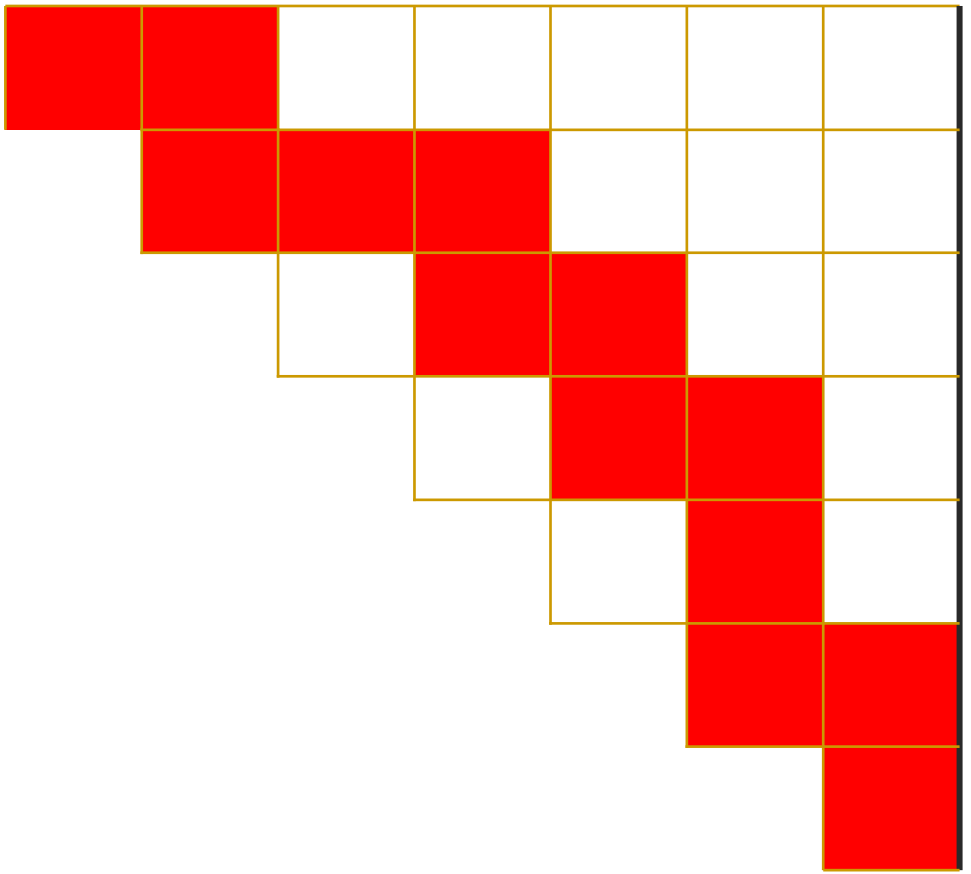
# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**



# Behavior of algorithm RS

**border points**
**inner points**
**skeletal points**

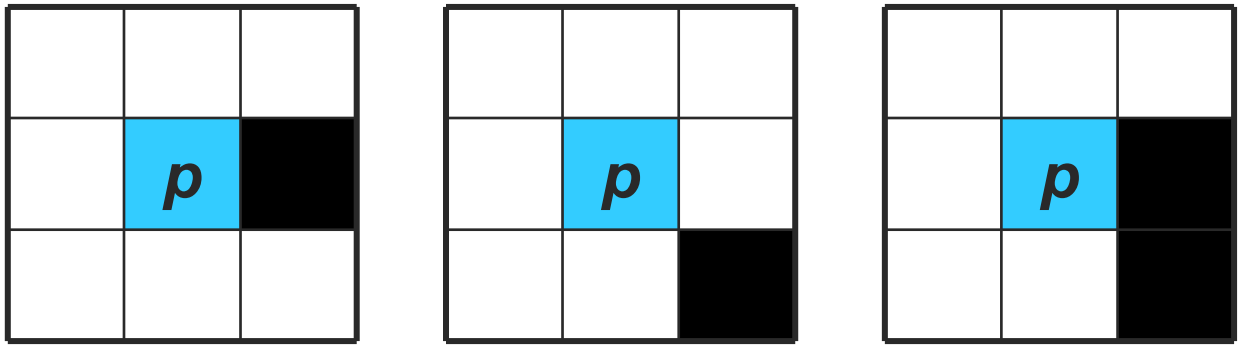


# Expected properties - algorithm RS

- Order-independency ✓
- Topology preserving ✓
- 1-pixel thin ✗
- Shape preserving ?

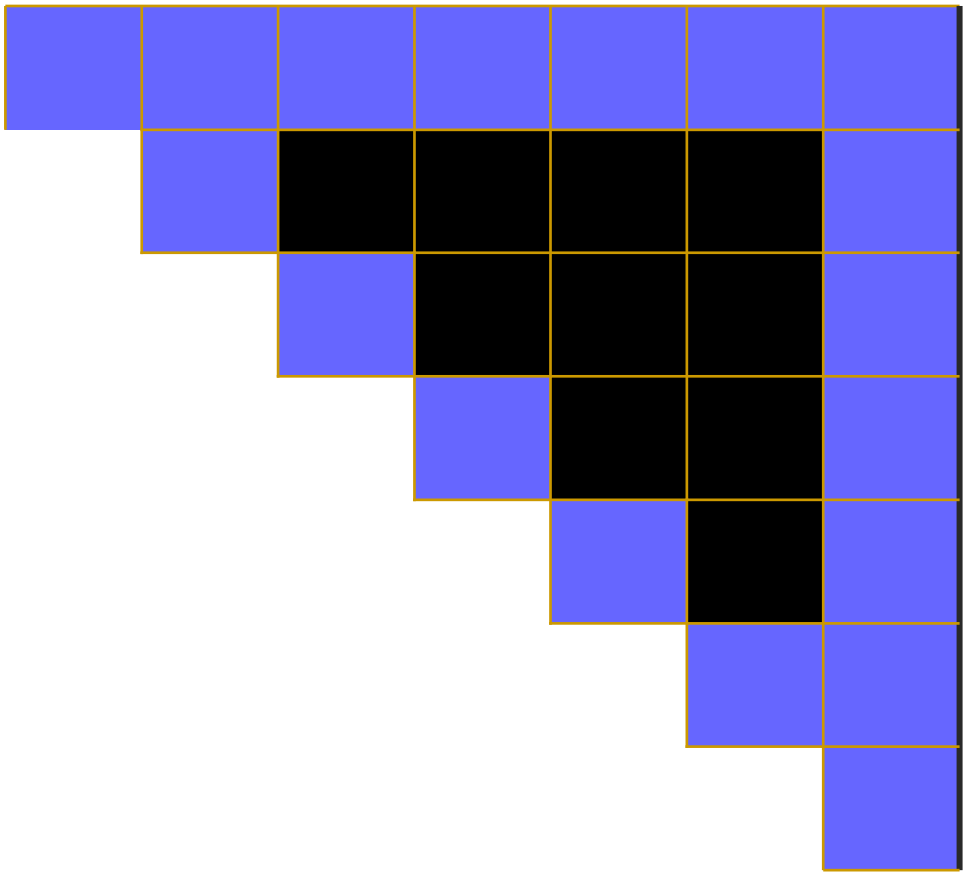
# What is an end-point?

- Different possible definitions
- We use the following supports and their $k$·90° rotations:



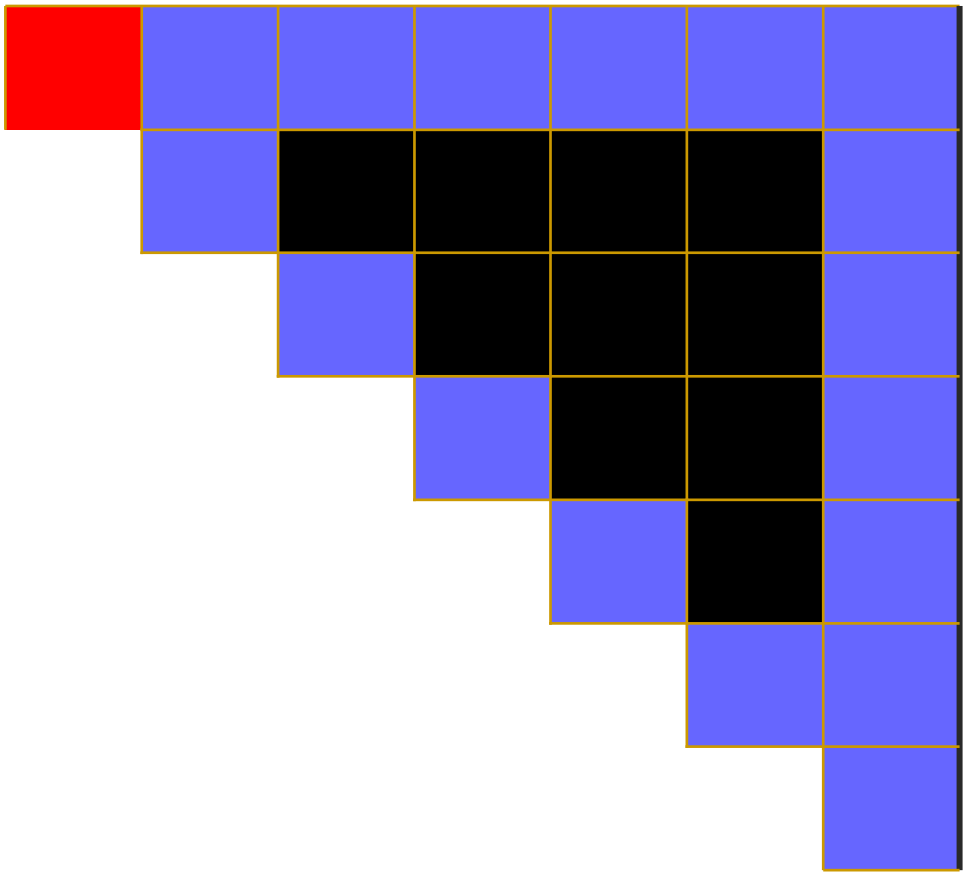
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



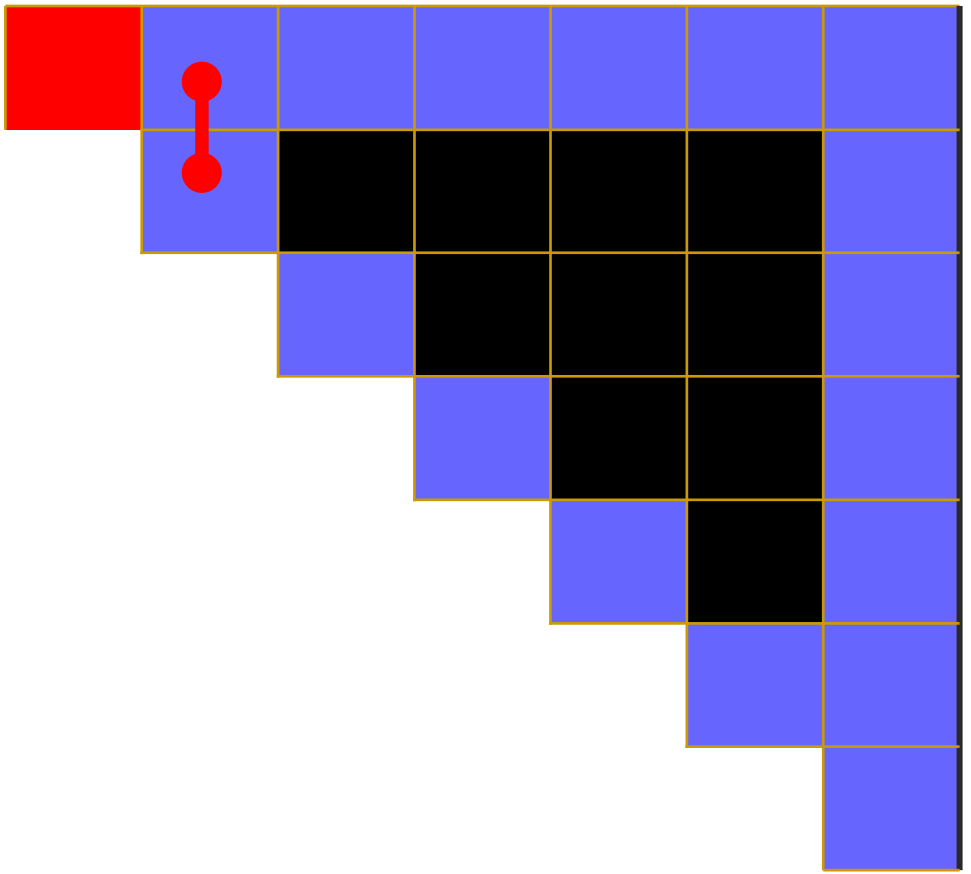
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



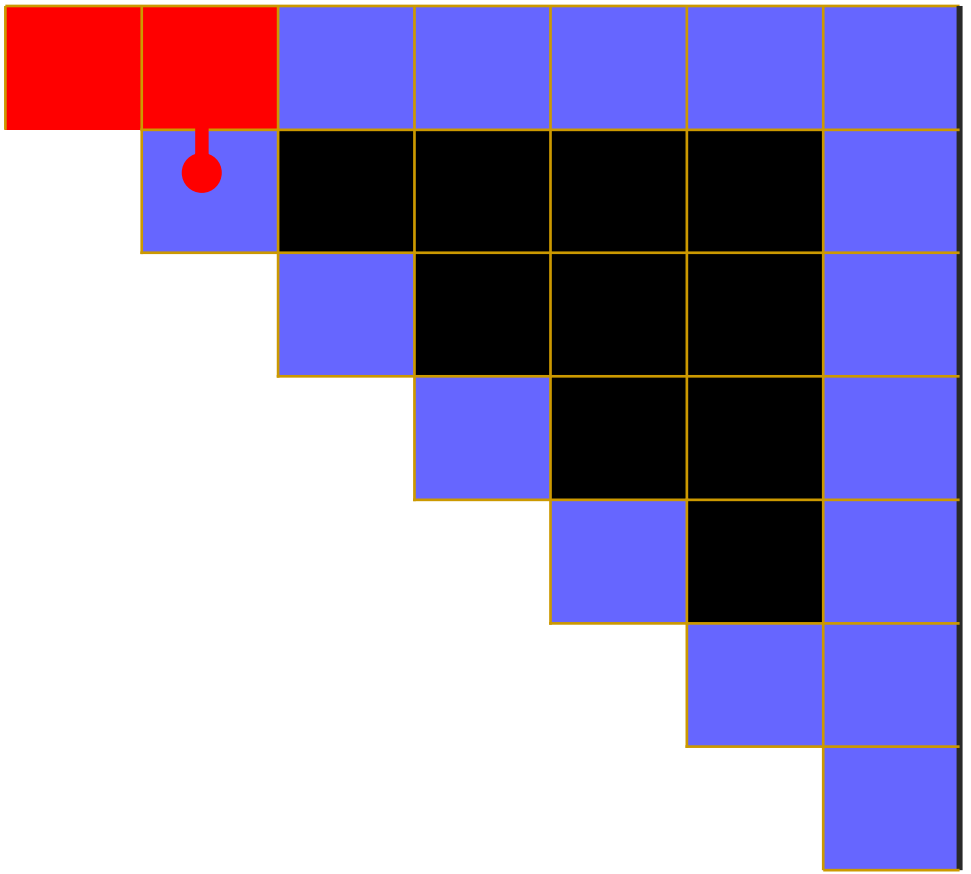
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



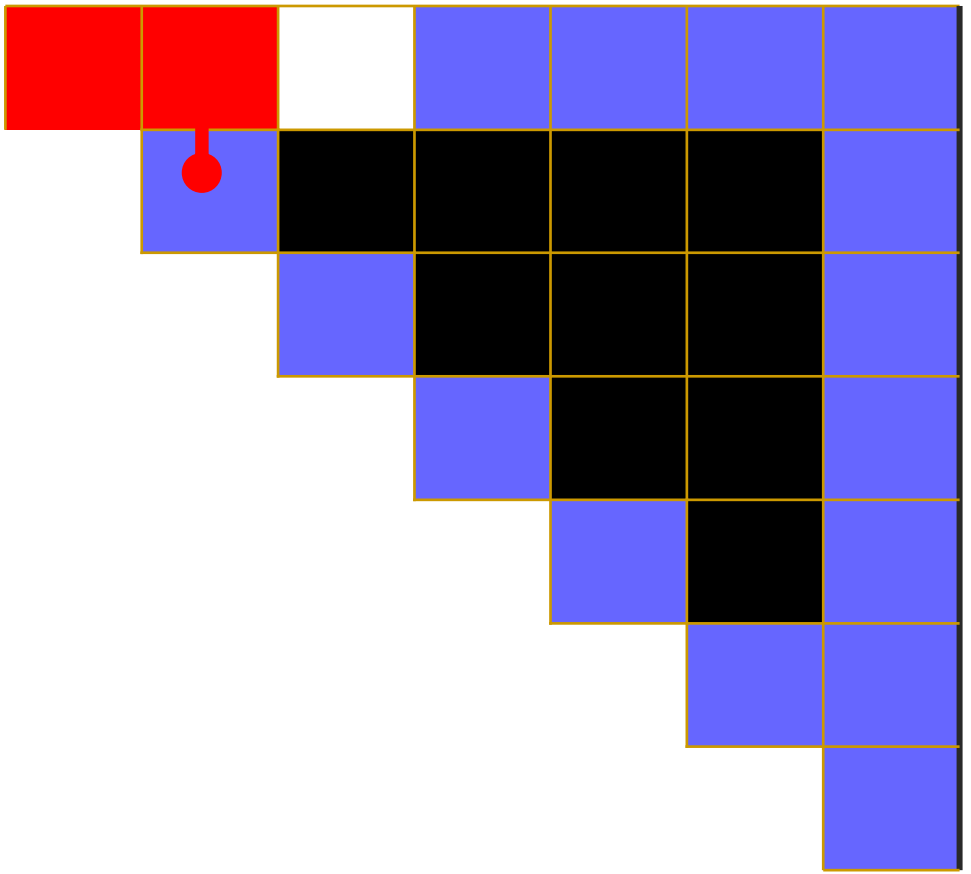
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



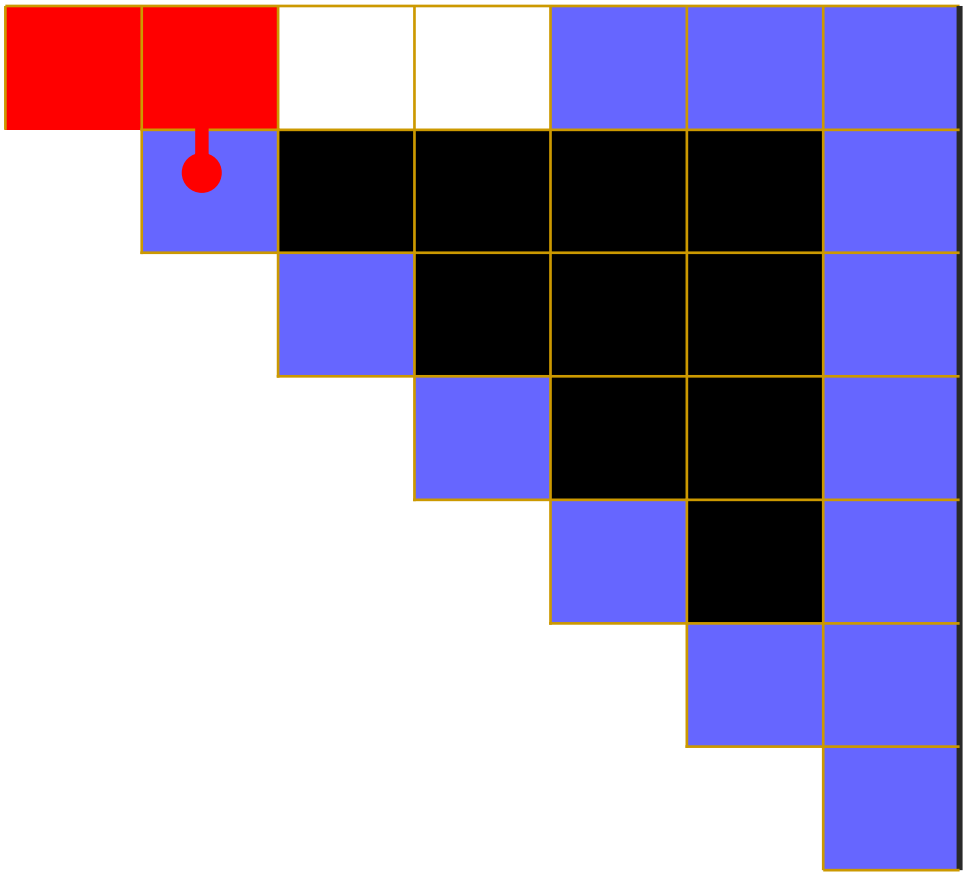
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



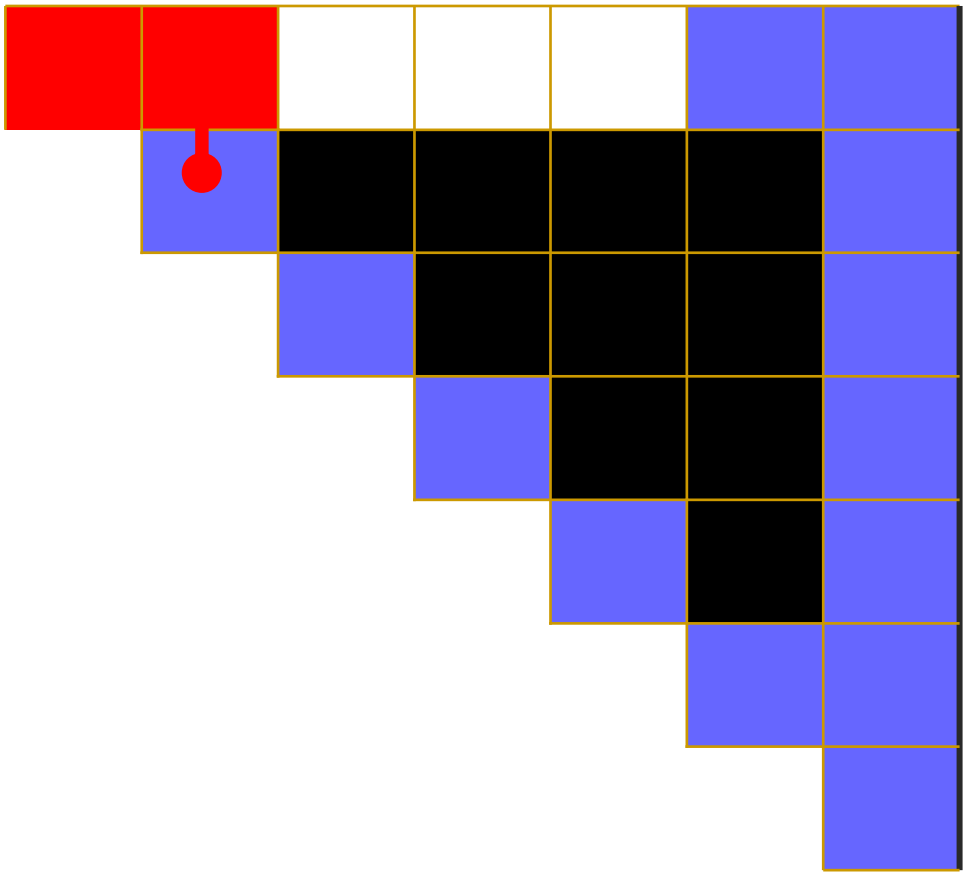
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



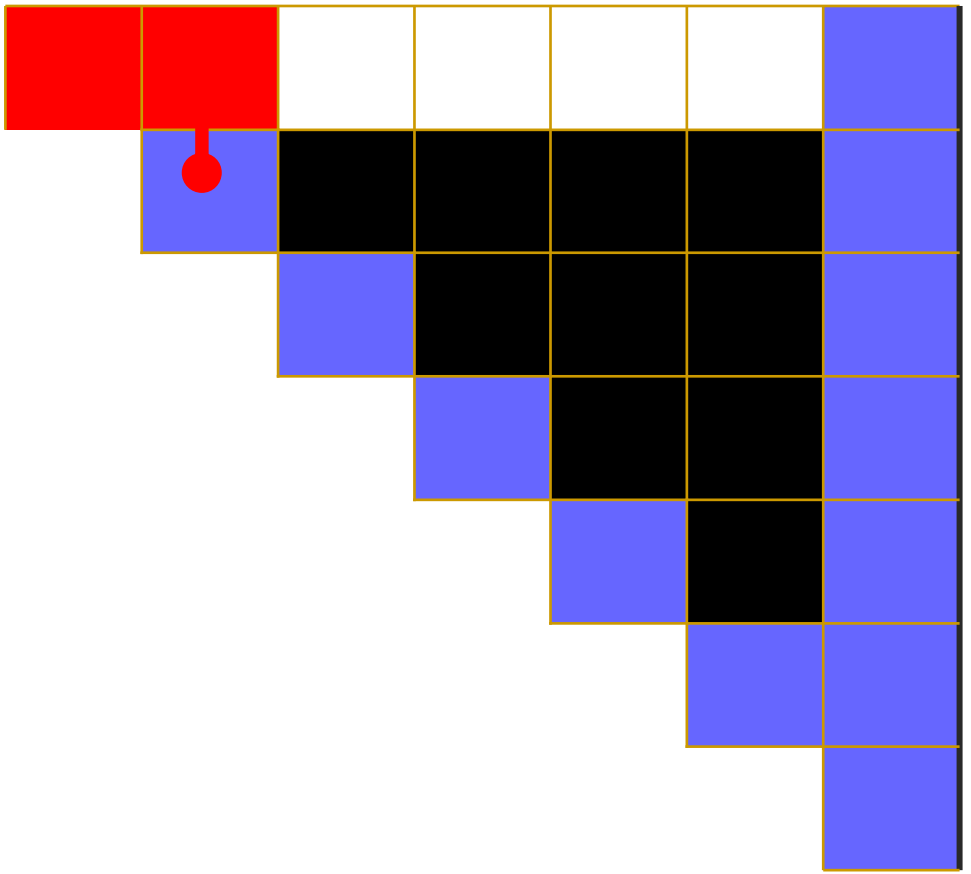
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



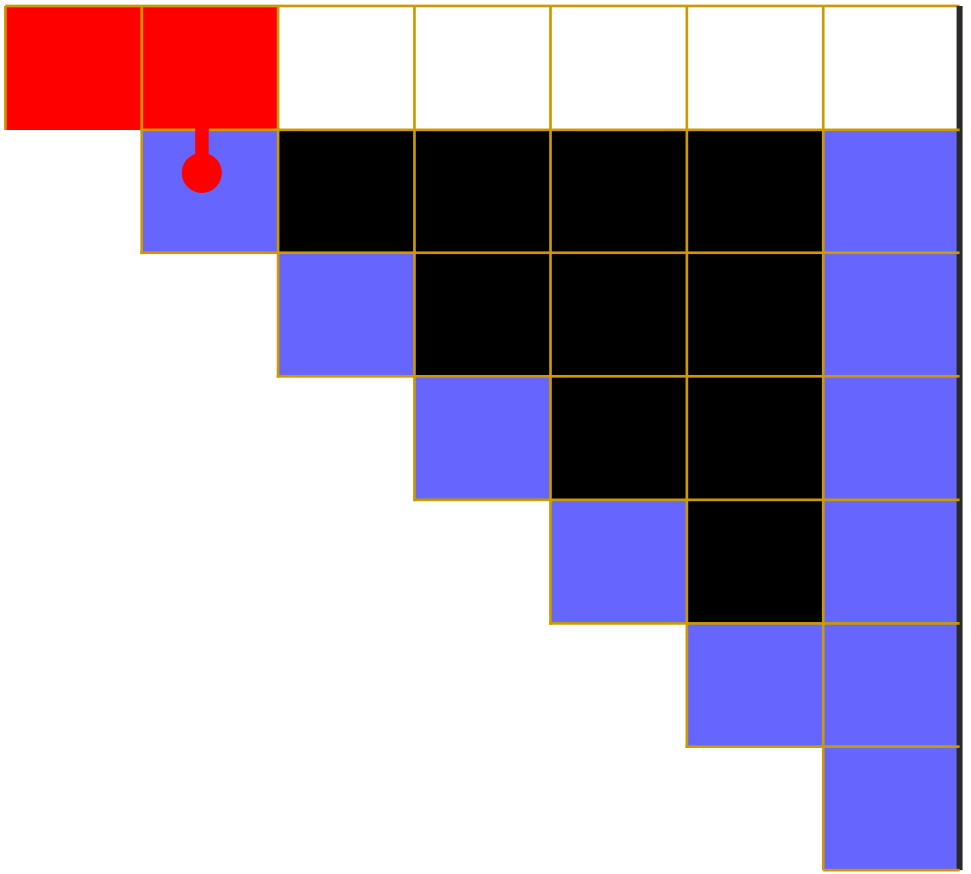
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



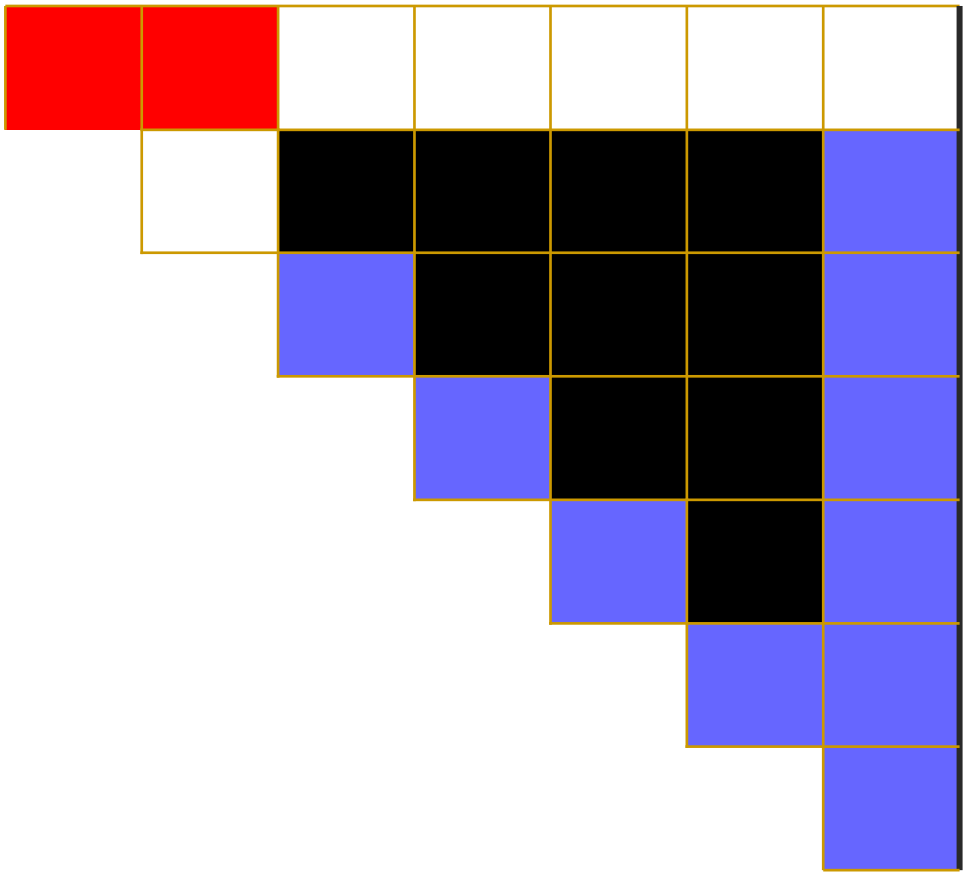
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



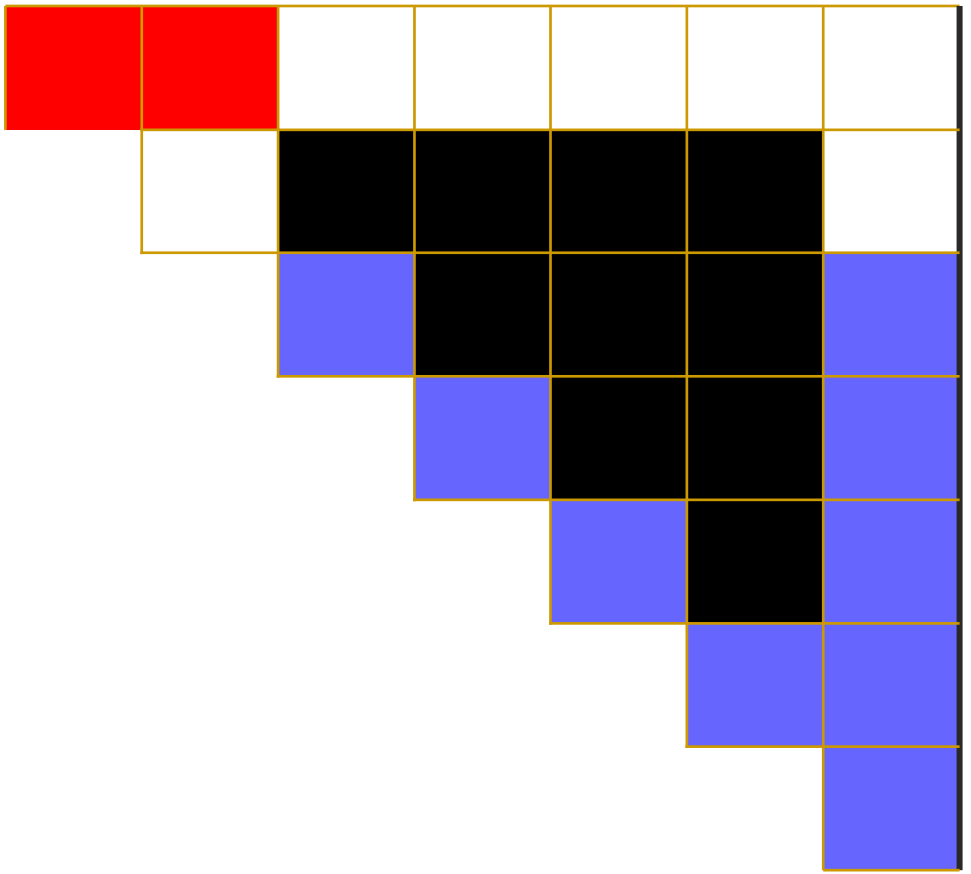
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



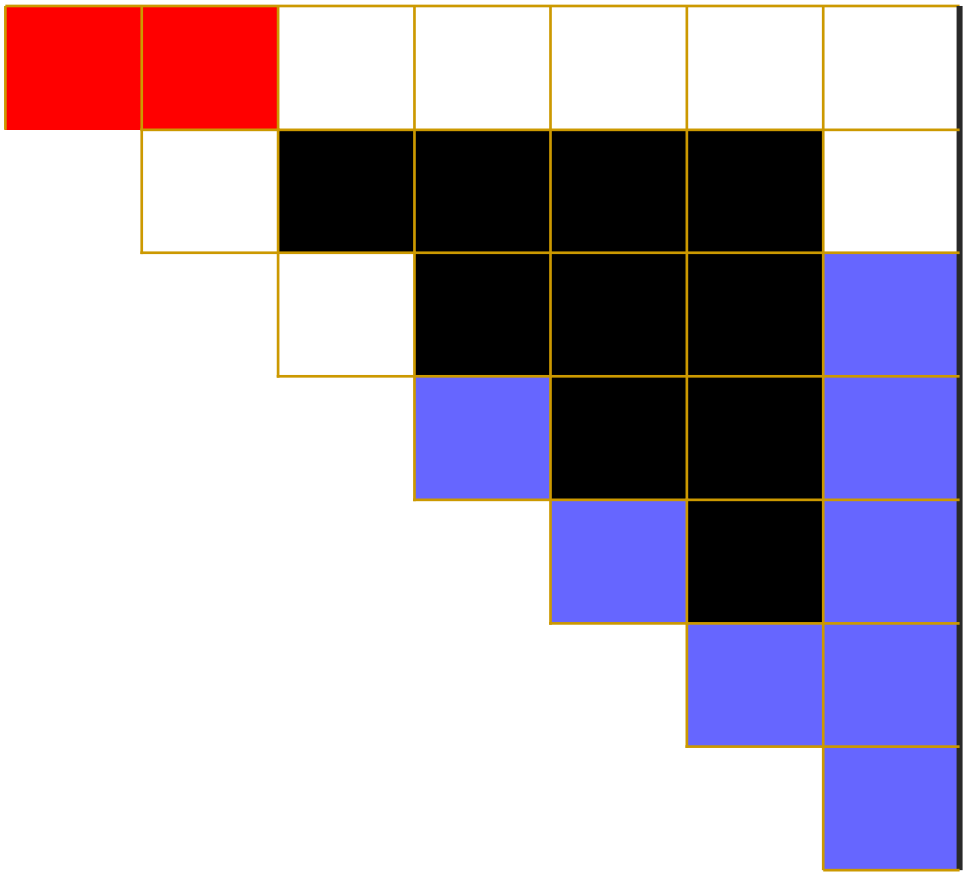
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



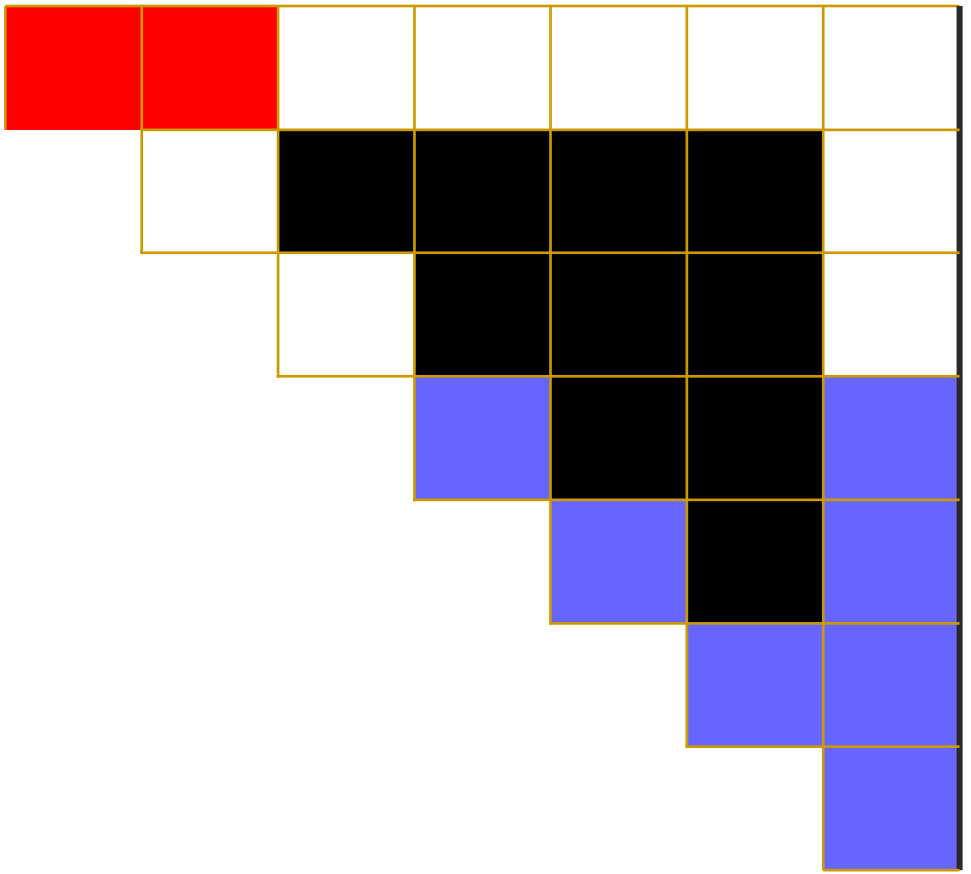
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



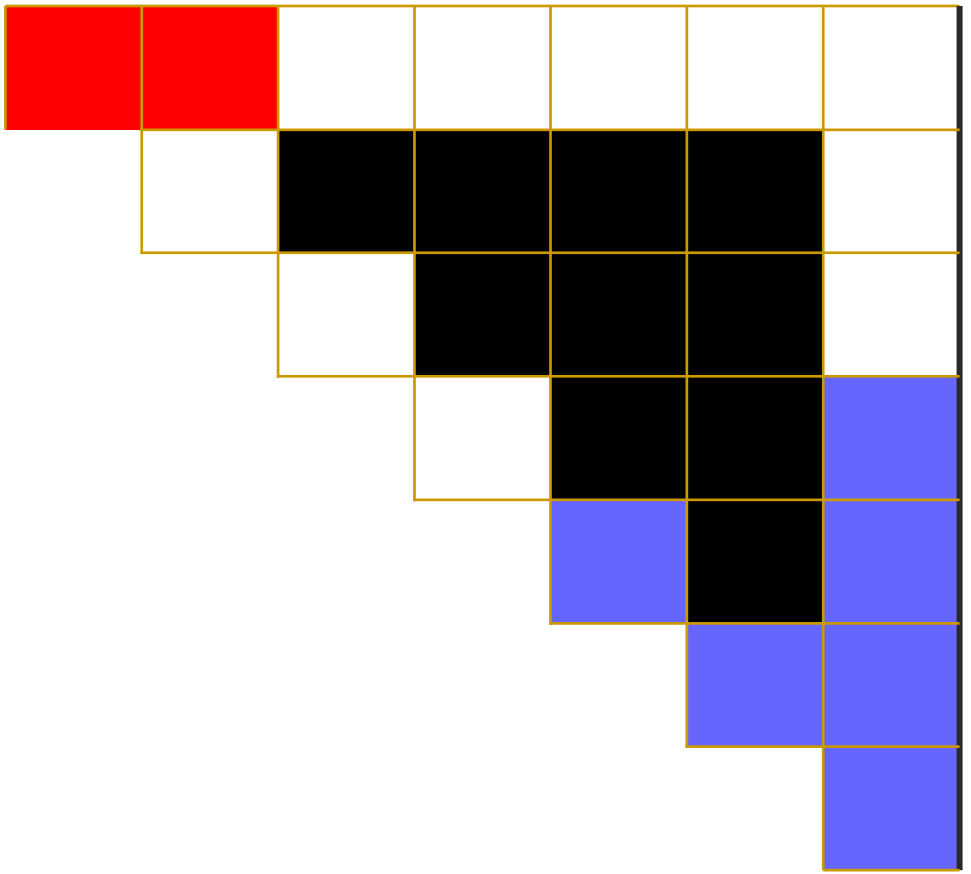
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



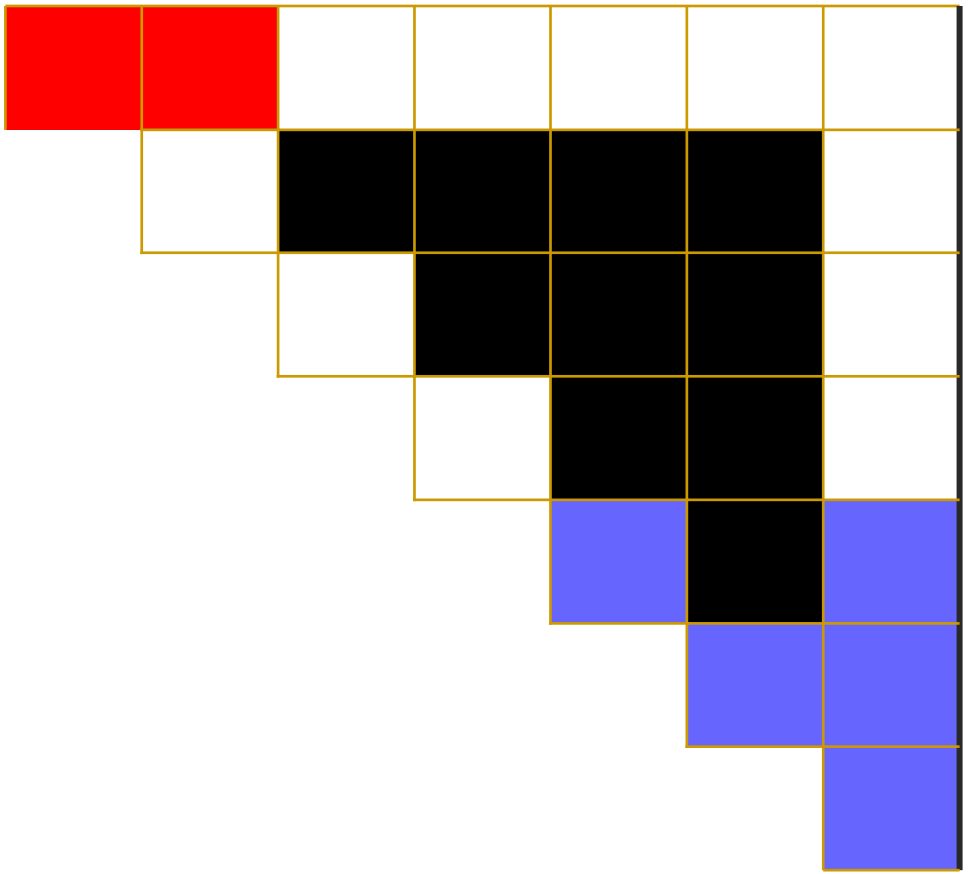
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



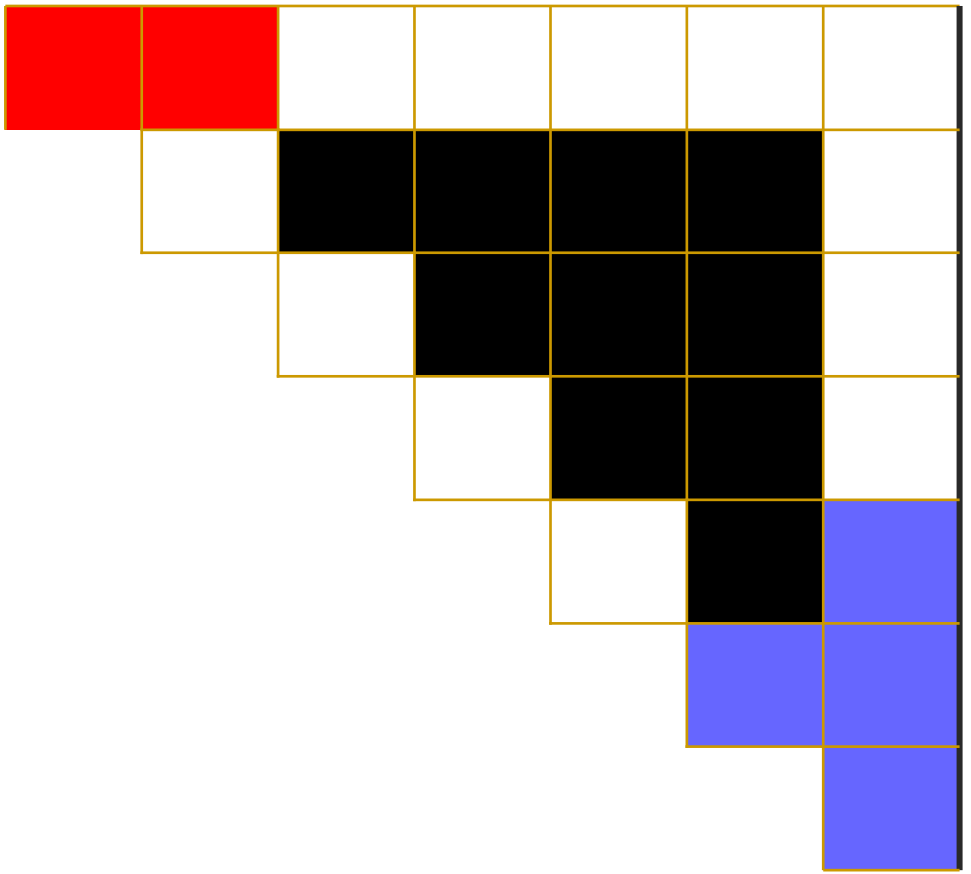
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



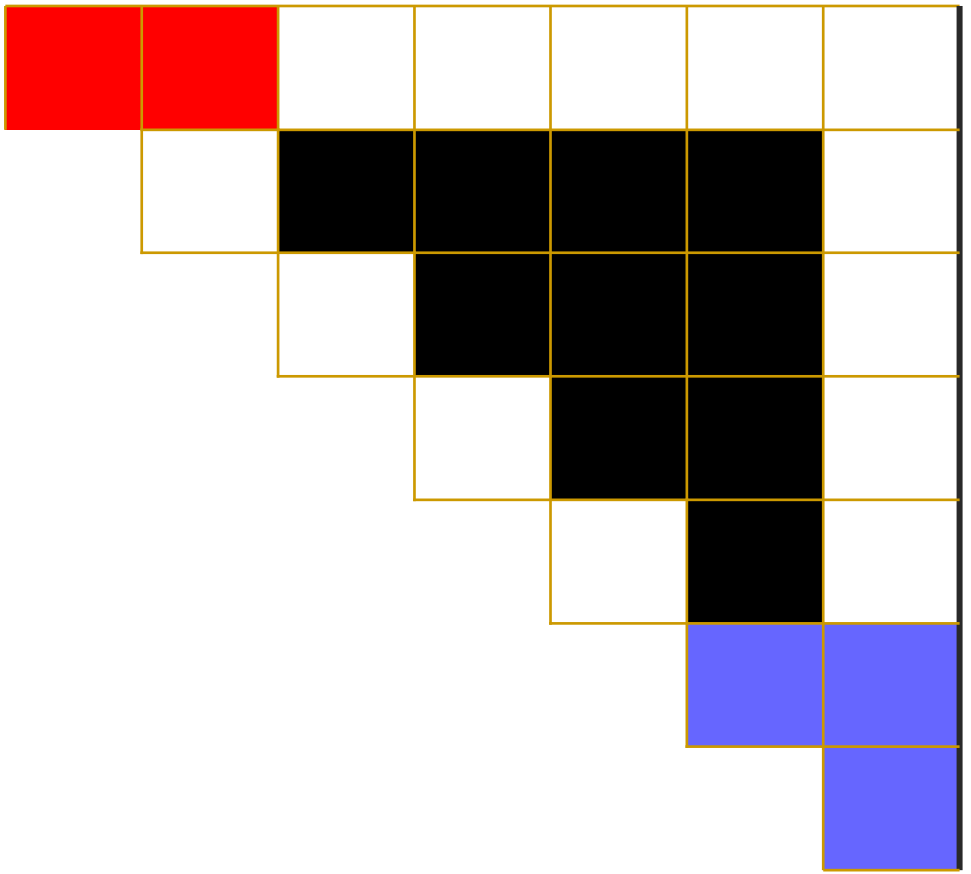
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



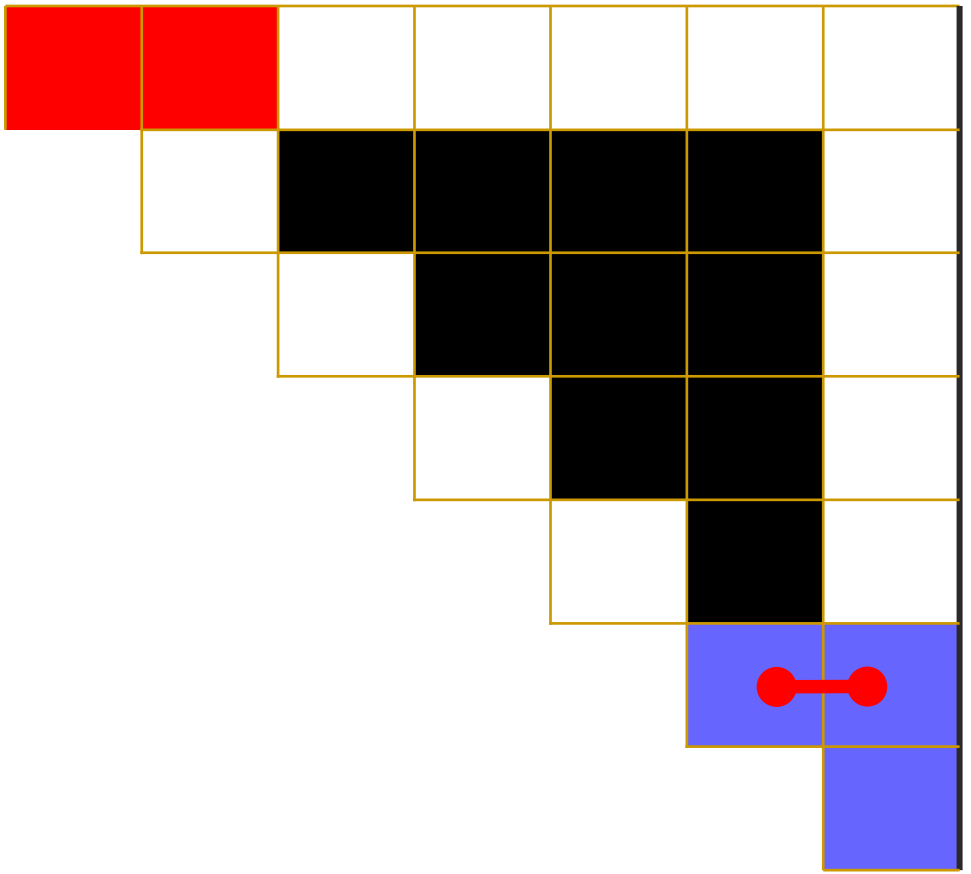
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



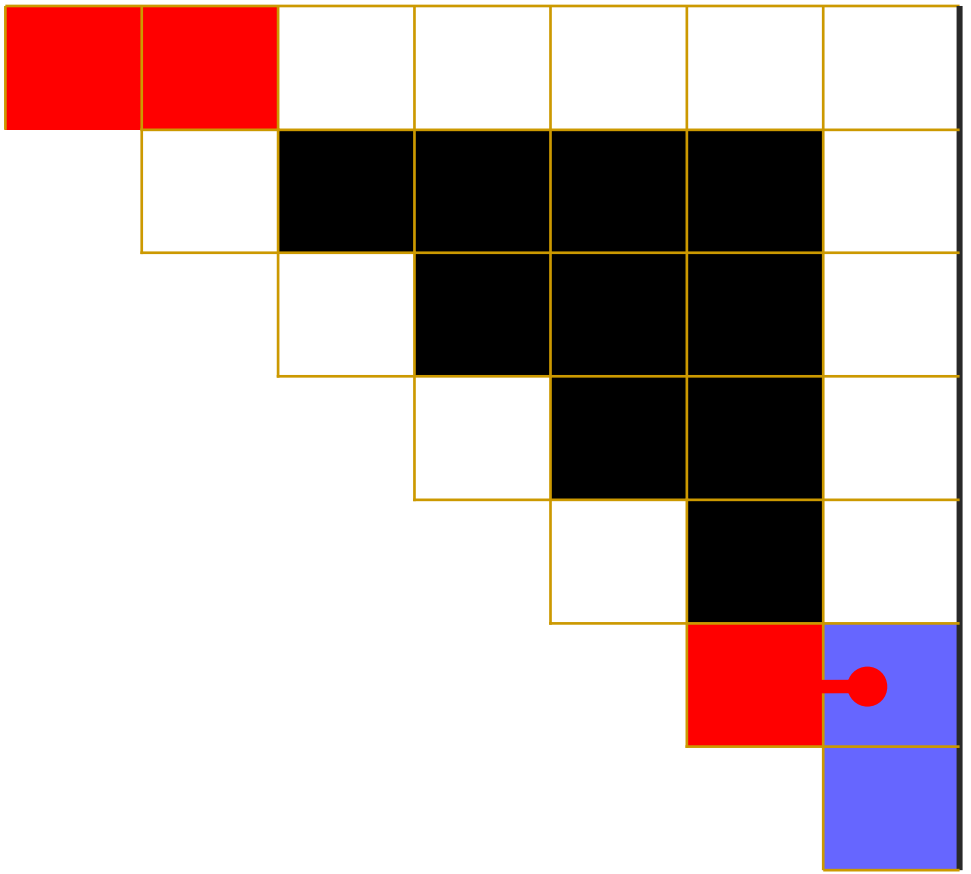
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



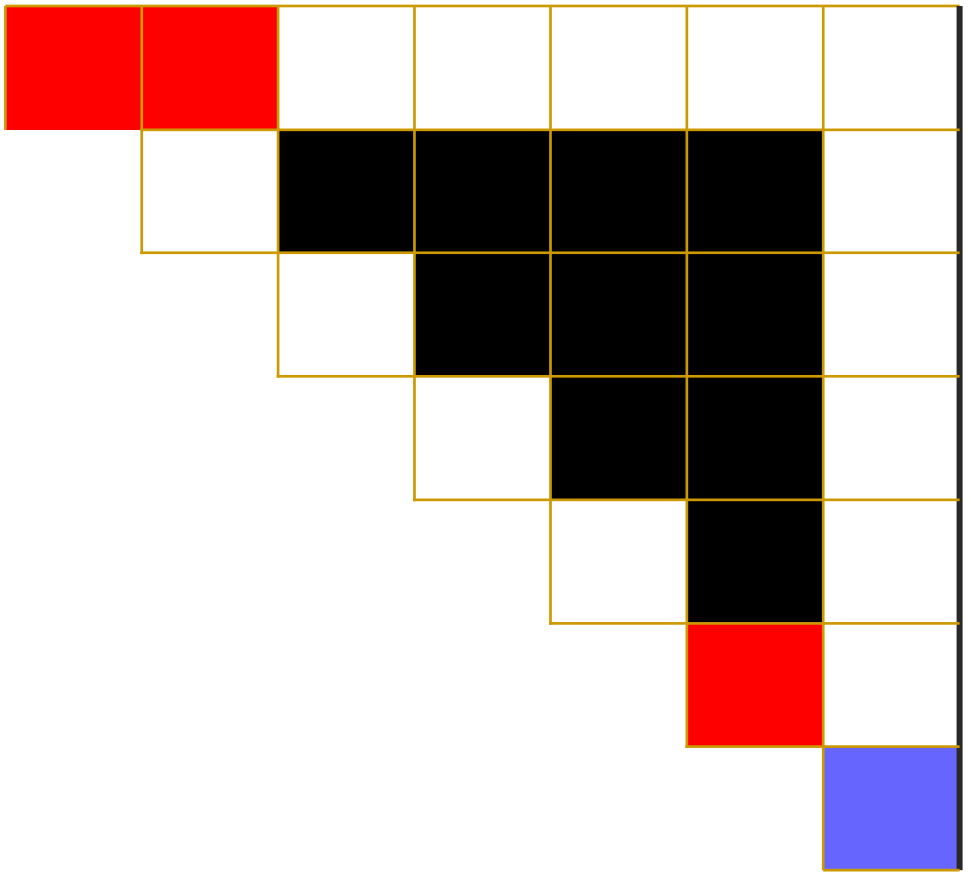
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



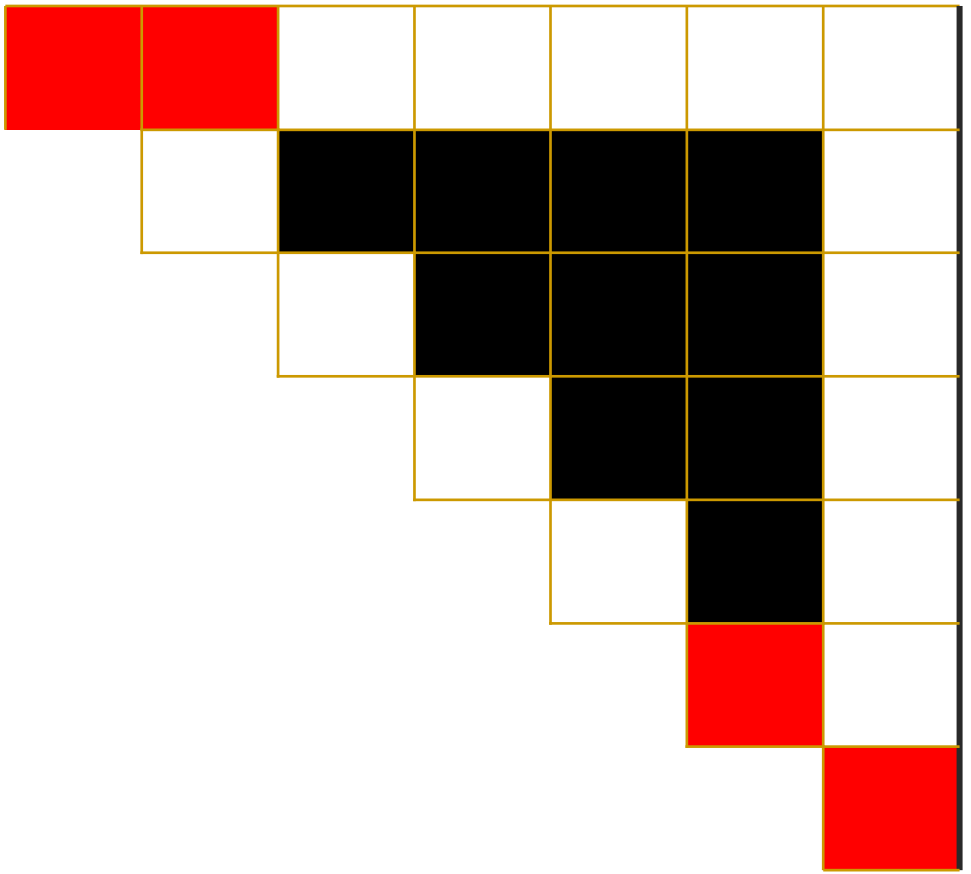
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



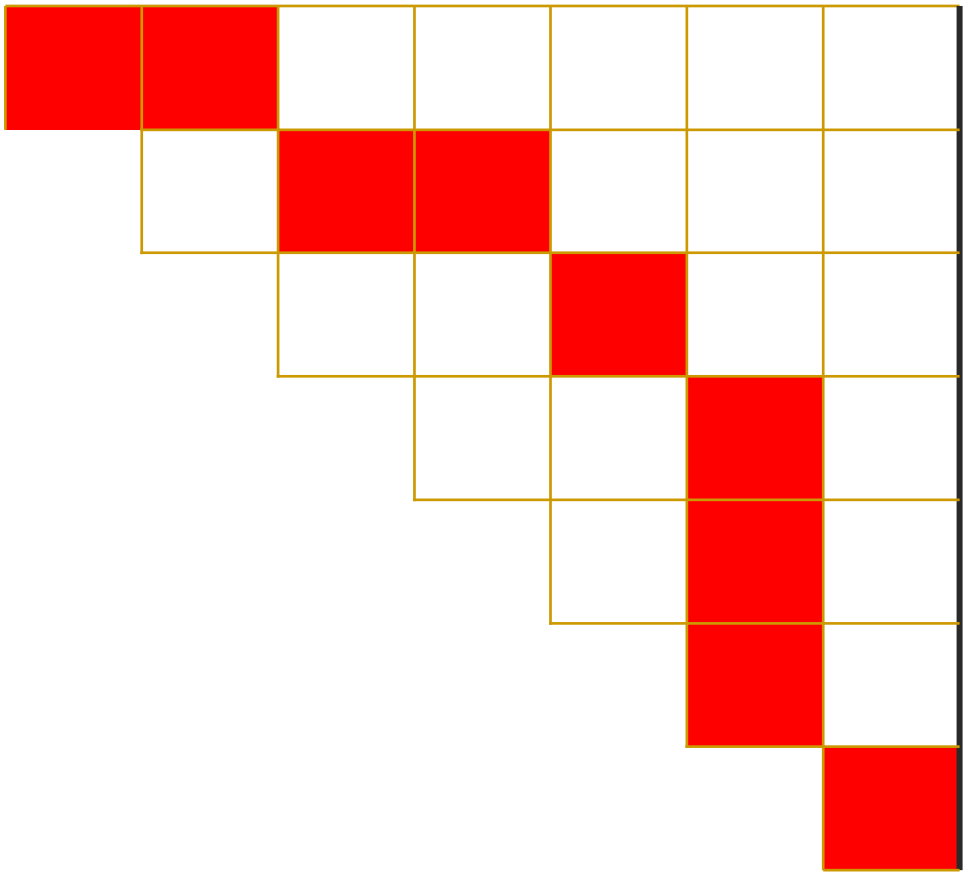
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**



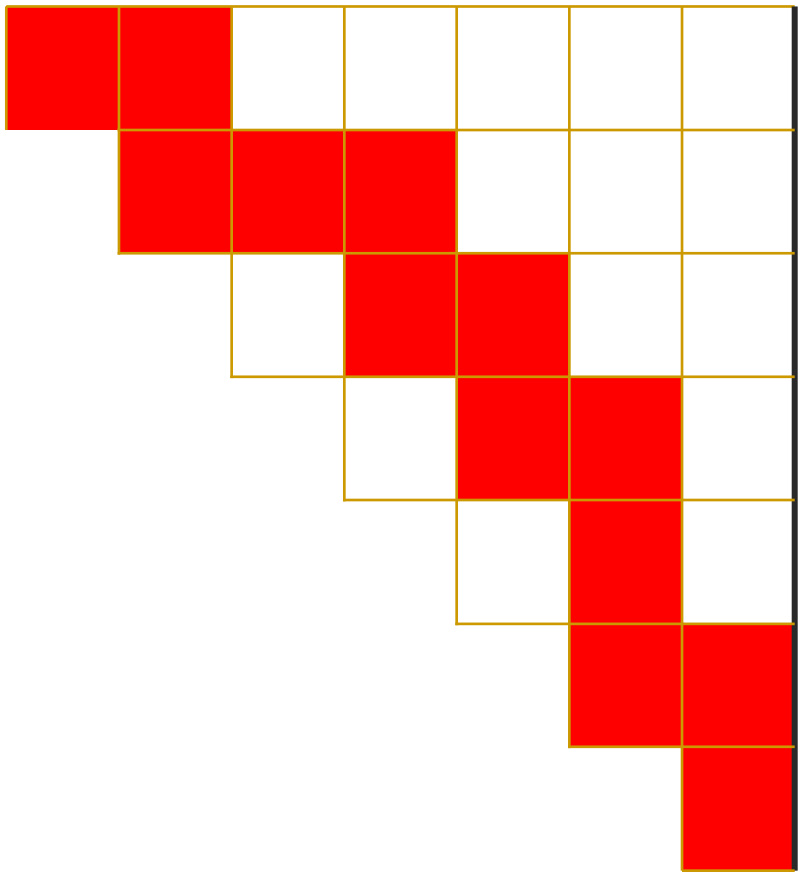
# Behavior of proposed algorithm

**border points**
**inner points**
**skeletal points**

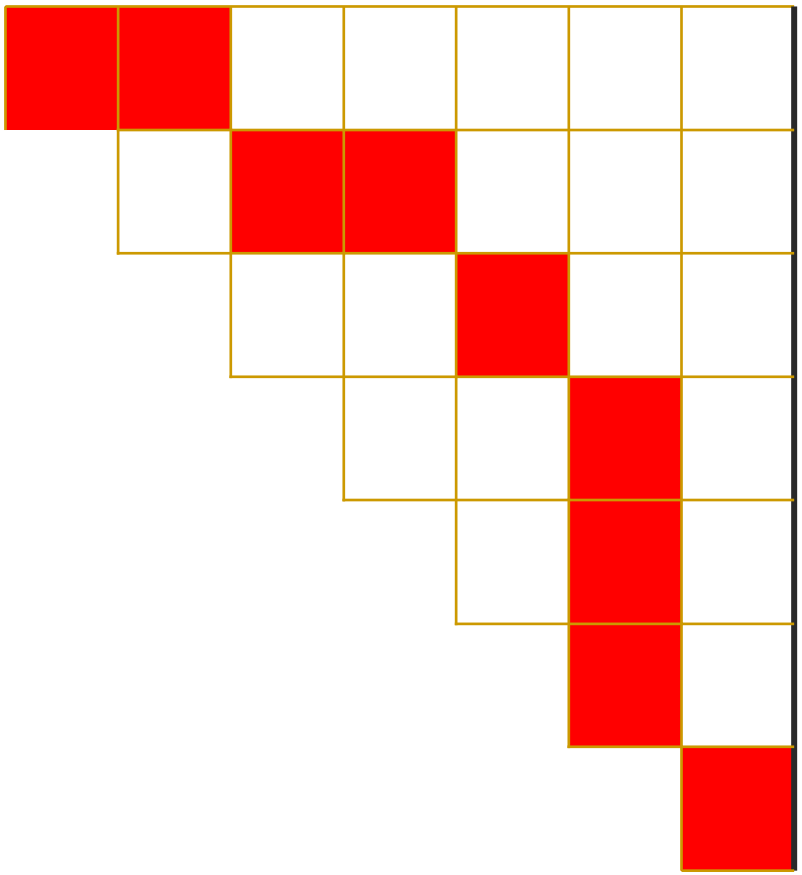


# Behavior of proposed algorithm

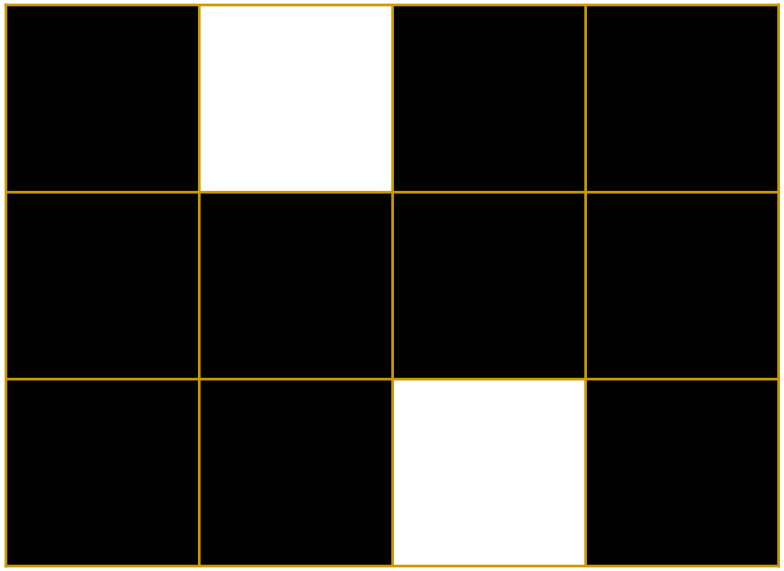
*Algorithm RS*



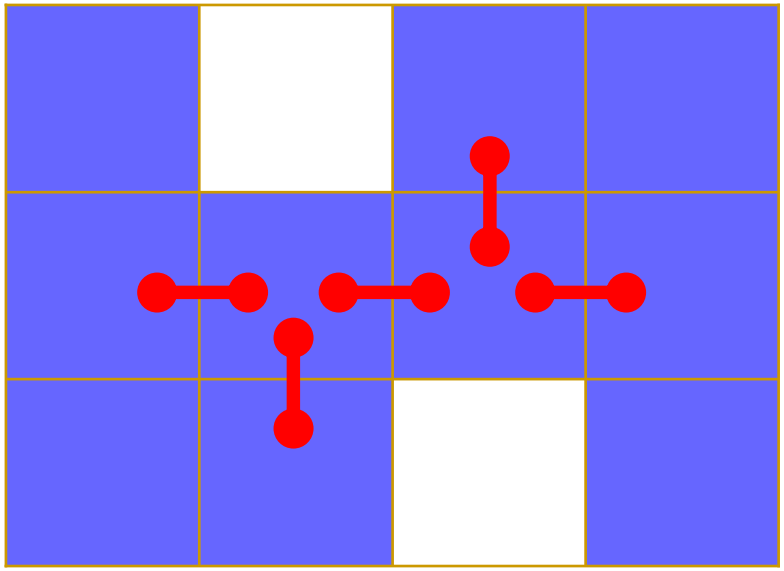
*Proposed algorithm*



# A more complicated case



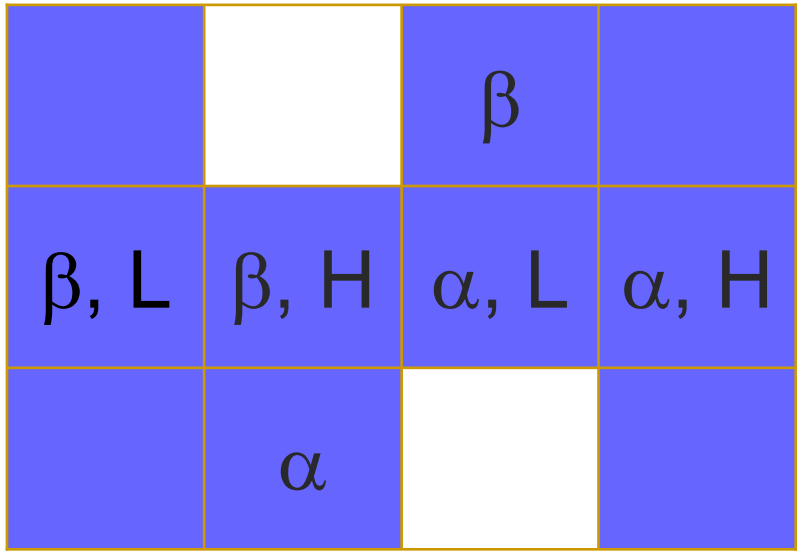
# A more complicated case



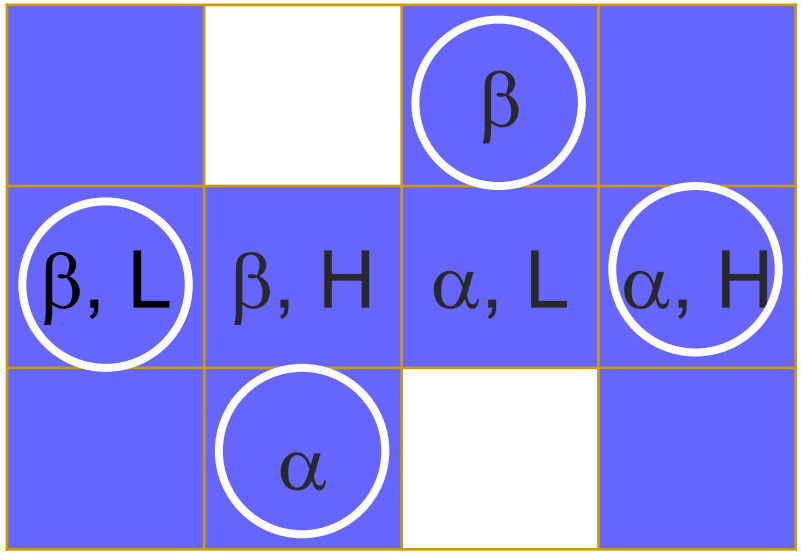
# How is order-independency guaranteed?

- Detect end-points in the first phase
- Precedency on different types of points in critical pairs:
  - $\alpha$-, $\beta$-, and $\gamma$-points
  - higher or lower index

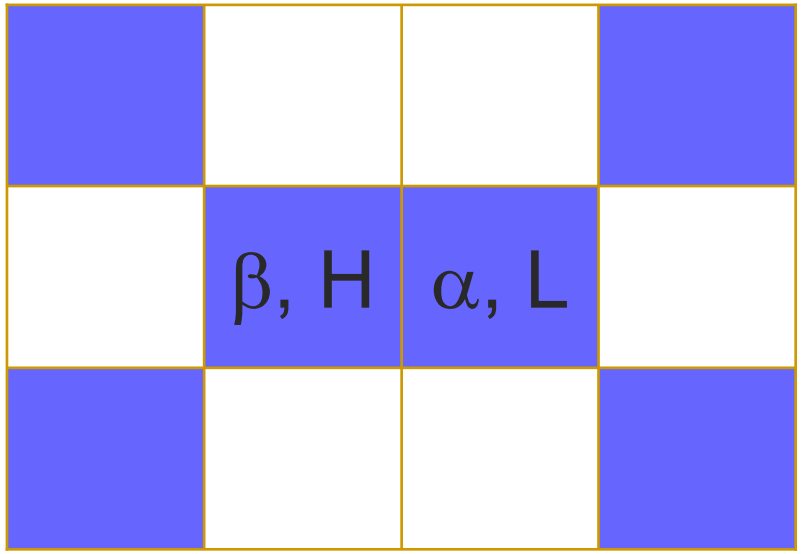
# Labeling the points of critical pairs



# Labeling the points of critical pairs



# Labeling the points of critical pairs



# Proposed algorithm

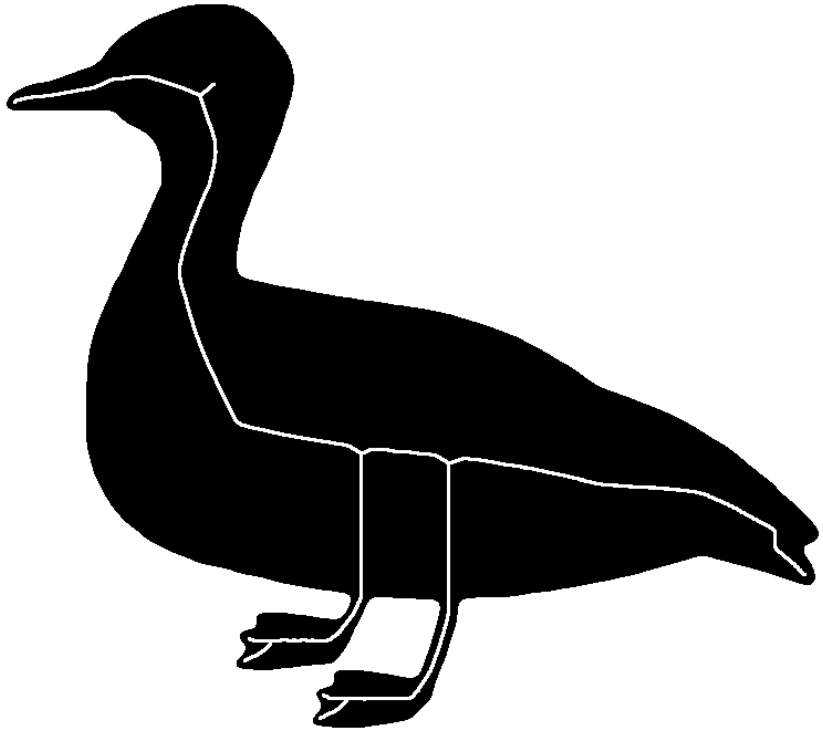
**Input:** array *P* containing the picture
**Output:** array *P* containing the skeleton

**repeat**
    *Phase 1:*
        detect end-points
        label *α*-, *β*-, *γ*- and corner points in *P*
    *Phase 2:*
        **for** each labeled point *p*
            **if** C(*p*)=1 **and** *p* is not an end-point **and**
                *p* is a preferred point
            **then**
                delete *p*
**until** no points are deleted

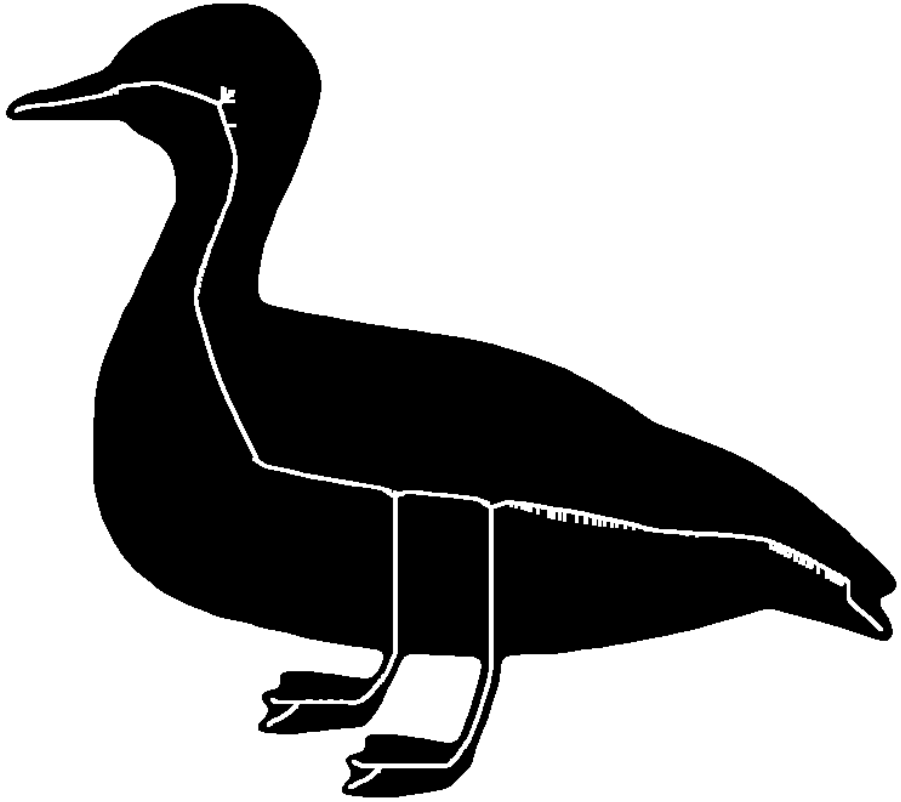
# Expected properties - proposed algorithm

- Order-independency ✓
- Topology preserving ✓
- 1-pixel thin ✓
- Shape preserving ✓

# Results

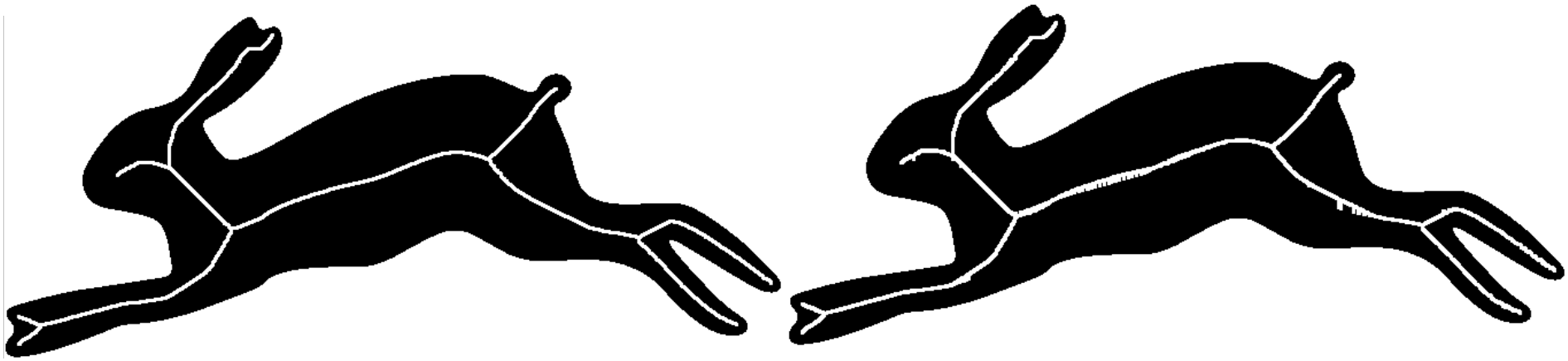


*Proposed algorithm*
(1590)



*Algorithm RS*
(2740)

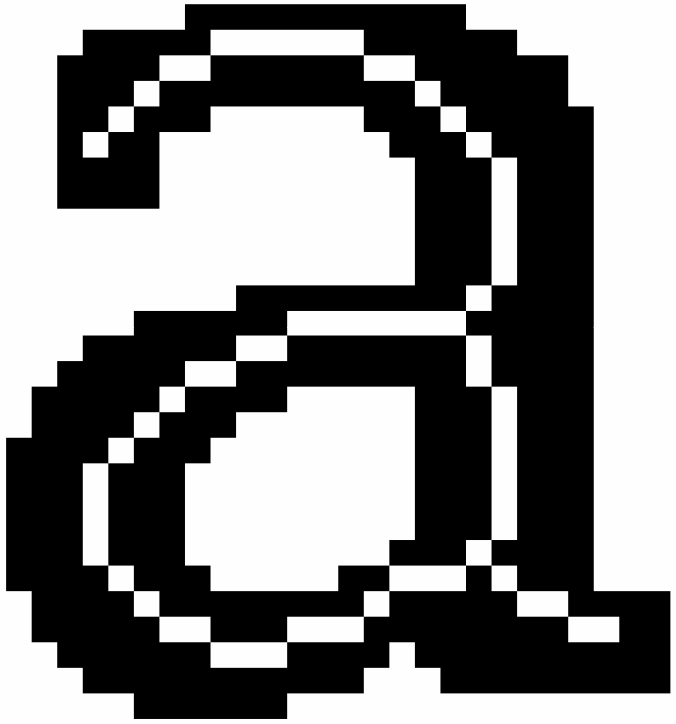
# Results



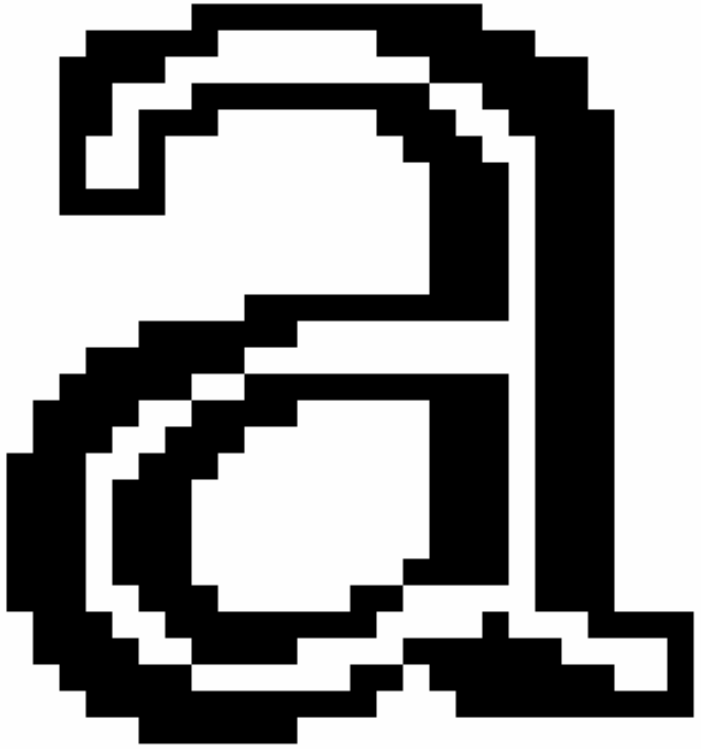
*Proposed algorithm*
(1085)

*Algorithm RS*
(1781)

# Results

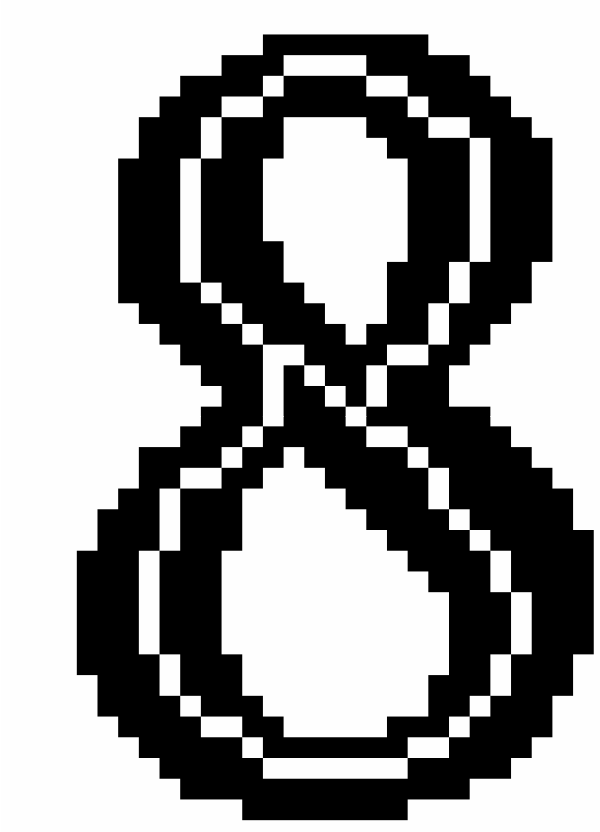


*Proposed algorithm*

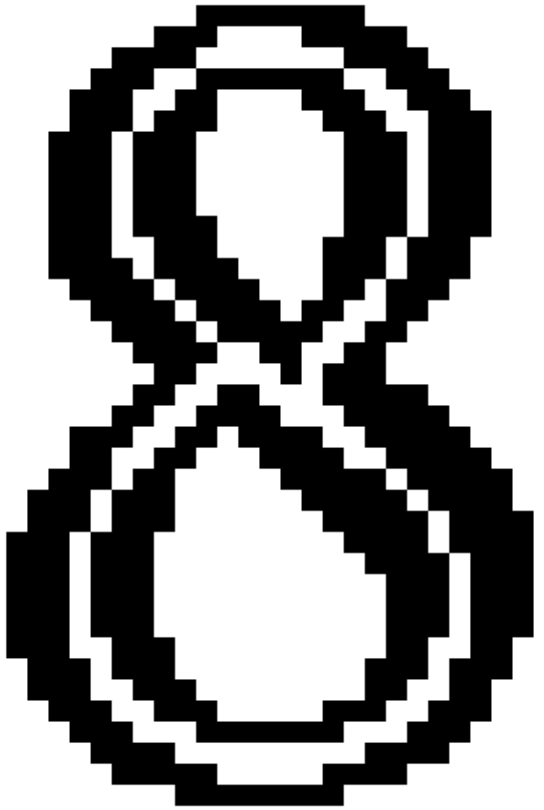


*Algorithm RS*

# Results



*Proposed algorithm*



*Algorithm RS*

# Thank you for your attention!