

Algorithms and Datastructures

Graphs, Depth-/Breadth-first Search, Graph-Connectivity

Prof. Dr. Rolf Backofen

Bioinformatics Group / Department of Computer Science

Algorithms and Datastructures, January 2017

Structure

Graphs

- Introduction

- Implementation

- Application example

Graphs

Introduction

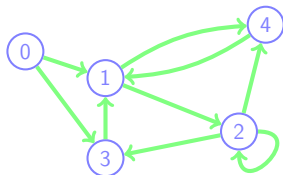
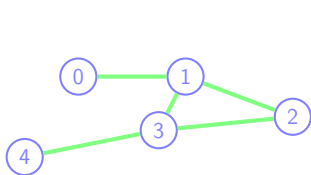
Graphs - Overview:

- ▶ Besides arrays, lists and trees the most common datastructure (Trees are a special type of graph)
- ▶ Representation of graphs in the computer
- ▶ Breadth first search (BFS)
- ▶ Depth first search (DFS)
- ▶ Connected components of a graph

Graphs

Introduction

Terminology:

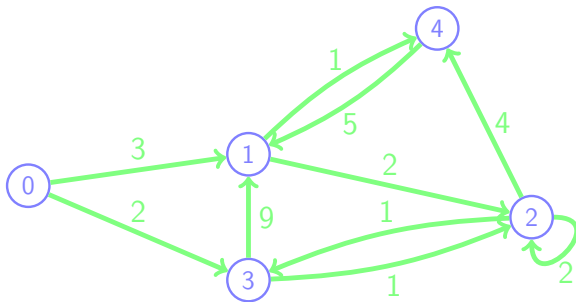


- ▶ Each Graph $G = (V, E)$ consists of:
 - ▶ A set of vertices (nodes) $V = \{v_1, v_2, \dots\}$
 - ▶ A set of edges (arcs) $E = \{e_1, e_2, \dots\}$
- ▶ Each edge connects two vertices ($u, v \in V$)
 - ▶ Undirected edge: $e = \{u, v\}$ (set)
 - ▶ Directed edge: $e = (u, v)$ (tuple)
- ▶ Self-loops are also possible: $e = (u, u)$ or $e = \{u, u\}$

Graphs

Introduction

Weighted graph:



- ▶ Each edge is marked with a real number named **weight**
- ▶ The **weight** is also named **length** or **cost** of the edge depending on the application

Graphs

Introduction

Example: Road network

- Intersections: **vertices**
- Roads: **edges**
- Travel time:
costs of the edges

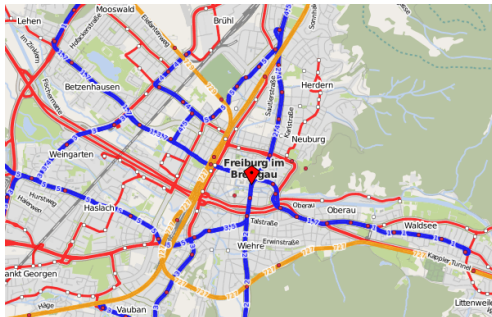


Figure: Map of Freiburg © OpenStreetMap

Graphs

Implementation

How to represent this graph computationally?

1. **Adjacency matrix** with space consumption $\Theta(|V|^2)$



Figure: Weighted graph with
 $|V| = 4$, $|E| = 6$

| | | end-vertex | | | |
|--------------|---|------------|---|----|----|
| | | 0 | 1 | 2 | 3 |
| start-vertex | 0 | | 2 | | 3 |
| | 1 | | | 9 | |
| | 2 | | | | -1 |
| | 3 | | 7 | -2 | |

Figure: Adjacency matrix

Graphs

Implementation

How to represent this graph computationally?

2. **Adjacency list / fields** with space consumption $\Theta(|V| + |E|)$
Each list item stores the **target vertex** and the **cost** of the edge

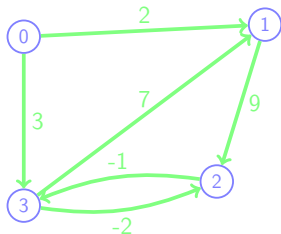


Figure: Weighted graph with
 $|V| = 4$, $|E| = 6$

| | | | |
|--------------|---|-------|-------|
| start-vertex | 0 | 1, 2 | 3, 3 |
| | 1 | 2, 9 | |
| | 2 | 3, -1 | |
| | 3 | 1, 7 | 2, -2 |

Figure: Adjacency list

Graphs

Implementation

Graph: Arrangement

- ▶ Graph is fully defined through the [adjacency matrix / list](#)
- ▶ The arrangement is not relevant for visualisation of the graph



Figure: Weighted graph with
 $|V| = 4$, $|E| = 6$



Figure: Same graph ordered by number -
outer planar graph

Graphs

Implementation - Python

```
class Graph:
    def __init__(self):
        self.vertices = []
        self.edges = []

    def addVertice(self, vert):
        self.vertices.append(vert)

    def addEdge(self, fromVert, toVert, cost):
        self.edges.append( \
            (fromVert, toVert, cost))

    ...
```

Graphs

Degrees (Valency)

Degree of a vertex: Directed graph: $G = (V, E)$

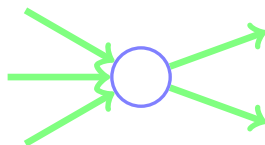


Figure: Vertex with in- / outdegree of 3 / 2

- **Indegree** of a vertex u is the number of **edge head ends** adjacent to the vertex

$$\deg^+(u) = |\{(v, u) : (v, u) \in E\}|$$

- **Outdegree** of a vertex u is the number of **edge tail ends** adjacent to the vertex

$$\deg^-(u) = |\{(u, v) : (u, v) \in E\}|$$

Graphs

Degrees (Valency)

Degree of a vertex: Undirected graph: $G = (V, E)$



Figure: Vertex with degree of 4

- **Degree** of a vertex u is the number of **vertices** adjacent to the vertex

$$\deg(u) = |\{\{v, u\} : \{v, u\} \in E\}|$$

Graphs

Paths

Paths in a graph: $G = (V, E)$



Figure: Undirected path of length 3
 $P = (0, 3, 2, 4)$



Figure: Directed path of length 3
 $P = (0, 3, 1, 4)$

- ▶ A path of G is a sequence of edges $u_1, u_2, \dots, u_i \in V$ with
 - ▶ Undirected graph: $\{u_1, u_2\}, \{u_2, u_3\}, \dots, \{u_{i-1}, u_i\} \in E$
 - ▶ Directed graph: $(u_1, u_2), (u_2, u_3), \dots, (u_{i-1}, u_i) \in E$

Graphs

Paths

Paths in a graph: $G = (V, E)$



Figure: **Directed path** of length 3
 $P = (0, 3, 1, 4)$



Figure: **Weighted path** with cost 6
 $P = (2, 3, 1)$

- ▶ The **length of a path** is: (also costs of a path)
 - ▶ Without weights: **number of edges** taken
 - ▶ With weights: **sum of weights of edges** taken

Graphs

Paths

Shortest path in a graph: $G = (V, E)$



Figure: Shortest path from 0 to 2 with cost / distance $d(0, 2) = ?$

- The shortest path between two vertices u, v is the path $P = (u, \dots, v)$ with the shortest length $d(u, v)$ or lowest costs

Graphs

Paths

Shortest path in a graph: $G = (V, E)$



Figure: Shortest path from 0 to 2 with cost / distance $d(0, 2) = 6$
 $P = (0, 1, 4, 3, 2)$

- The **shortest path** between two vertices u, v is the path $P = (u, \dots, v)$ with the shortest length $d(u, v)$ or lowest costs

Graphs

Paths

Diameter of a graph: $G = (V, E)$

$$d = \max_{u,v \in V} d(u, v)$$

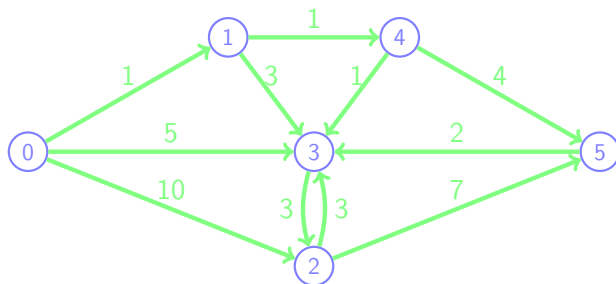


Figure: Diameter of graph is $d = ?$

- The **diameter** of a graph is the length / the costs of the longest shortest path

Graphs

Paths

Diameter of a graph: $G = (V, E)$

$$d = \max_{u,v \in V} d(u, v)$$



Figure: Diameter of graph is $d = 10$, $P = (3, 2, 5)$

- The **diameter** of a graph is the length / the costs of the longest shortest path

Graphs

Connected Components

Connected components: $G = (V, E)$



Figure: Three connected components

- ▶ Undirected graph:
 - ▶ All connected components are a partition of V

$$V = V_1 \cup \dots \cup V_k$$

- ▶ Two vertices u, v are in the same connected component if a path between u and v exists

Graphs

Connected Components

Connected components: $G = (V, E)$

- ▶ Directed graph:
 - ▶ Named **strongly connected components**
 - ▶ Direction of edge has to be regarded
 - ▶ Not part of this lecture

Graphs

Connected Components - Graph Exploration

Graph Exploration: (Informal definition)

- ▶ Let $G = (V, E)$ be a graph and $s \in V$ a start vertex
- ▶ We visit each reachable vertex connected to s
- ▶ **Breadth-first search:** in order of the smallest distance to s
- ▶ **Depth-first search:** in order of the largest distance to s
- ▶ Not a problem on its own but is often used as subroutine of other algorithms
 - ▶ Searching of connected components
 - ▶ Flood fill in drawing programmes

Graphs

Connected Components - Breadth-First Search

Breadth-First Search:

1. We start with all vertices unmarked and mark visited vertices
2. Mark the start vertex s (level 0)
3. Mark all unmarked connected vertices (level 1)
4. Mark all unmarked vertices connected to a level 1-vertex (level 2)
5. Iteratively mark reachable vertices for all levels
6. All connected nodes are now marked and in the same connected component as the start vertex s

Graphs

Connected Components - Breadth-First Search

- ▶ The marked vertices create a “spanning tree” containing all reachable nodes



Figure: spanning tree of a breadth-first search

Graphs

Connected Components - Breadth-First Search

- ▶ The marked vertices create a “spanning tree” containing all reachable nodes



Figure: spanning tree of a breadth-first search

Graphs

Connected Components - Breadth-First Search

- ▶ The marked vertices create a “spanning tree” containing all reachable nodes

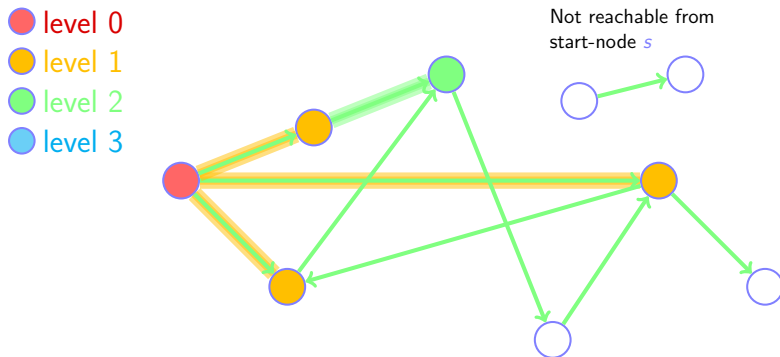


Figure: spanning tree of a breadth-first search

Graphs

Connected Components - Breadth-First Search

- ▶ The marked vertices create a “spanning tree” containing all reachable nodes



Figure: spanning tree of a breadth-first search

Graphs

Connected Components - Breadth-First Search

- ▶ The marked vertices create a “spanning tree” containing all reachable nodes



Figure: spanning tree of a breadth-first search

Graphs

Connected Components - Breadth-First Search

- ▶ The marked vertices create a “spanning tree” containing all reachable nodes



Figure: spanning tree of a breadth-first search

Graphs

Connected Components - Breadth-First Search

- ▶ The marked vertices create a “spanning tree” containing all reachable nodes



Figure: spanning tree of a breadth-first search

Graphs

Connected Components - Breadth-First Search

- ▶ The marked vertices create a “spanning tree” containing all reachable nodes



Figure: spanning tree of a breadth-first search

Graphs

Connected Components - Depth-First Search

Depth-First Search:

1. We start with all vertices unmarked and **mark visited vertices**
2. Mark the start vertex **s**
3. Pick an unmarked **connected vertex** and start a **recursive depth-first search** with the vertex as start vertex
(continue on step 2)
4. If no unmarked connected vertex exists go one vertex back and continue recursive search
(reduce the recursion level by one)

Graphs

Connected Components - Depth-First Search

Depth-first search:

- ▶ Search starts with **long paths** (searching with depth)
- ▶ Marks like **breadth-first search** all connected vertices
- ▶ If the graph is acyclic we get a **topological sorting**
 - ▶ Each newly visited vertex gets marked by an increasing number
 - ▶ The numbers increase with path length from the start vertex

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes

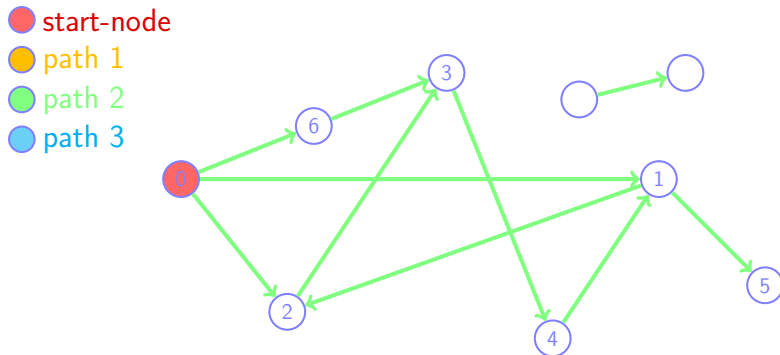


Figure: spanning tree of a depth-first search

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes

● start-node

● path 1

● path 2

● path 3

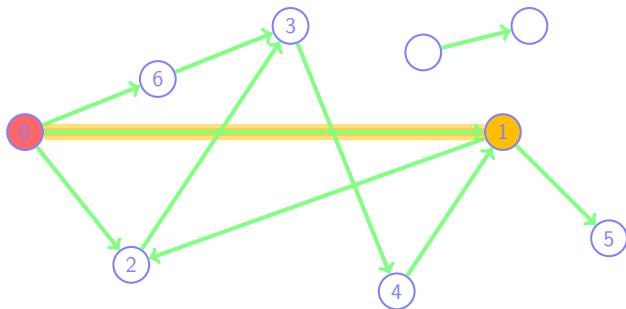


Figure: spanning tree of a depth-first search

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes



Figure: spanning tree of a depth-first search

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes

● start-node

● path 1

● path 2

● path 3

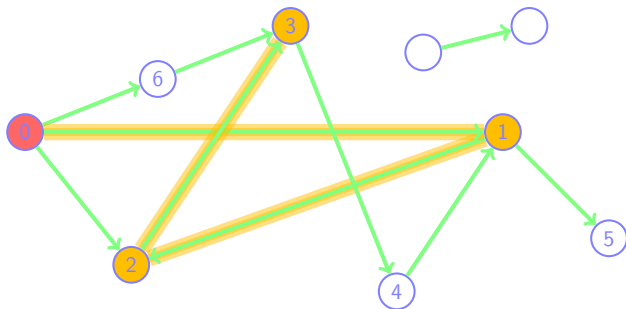


Figure: spanning tree of a depth-first search

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes

● start-node

● path 1

● path 2

● path 3



Figure: spanning tree of a depth-first search

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes

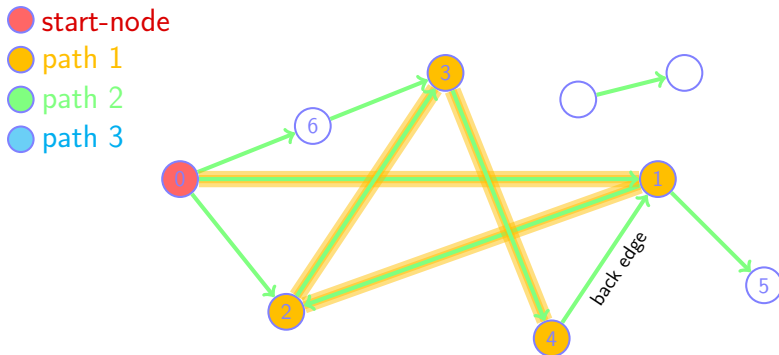


Figure: spanning tree of a depth-first search

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes

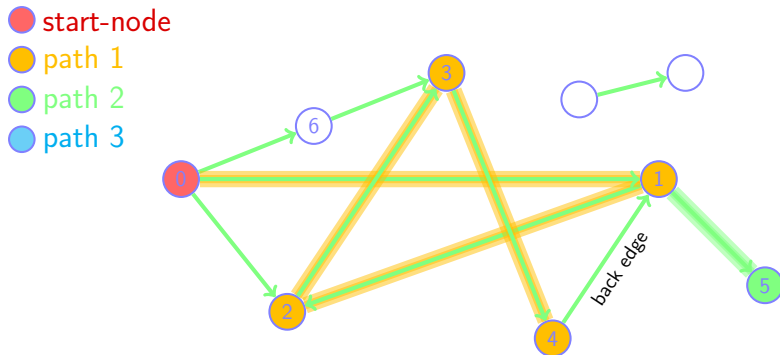


Figure: spanning tree of a depth-first search

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes



Figure: spanning tree of a depth-first search

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes



Figure: spanning tree of a depth-first search

Graphs

Connected Components - Depth-First Search

- ▶ The marked vertices create a different spanning tree containing all reachable nodes



Figure: spanning tree of a depth-first search

Graphs

Why is this called Breadth - and Depth First Search?

Graphs

Connected Components - Breadth-/Depth-First Search

Runtime complexity:

- ▶ Constant costs for each visited vertex and edge
- ▶ We get a runtime complexity of $\Theta(|V'| + |E'|)$
- ▶ Let V' and E' be the reachable vertices and edges
- ▶ All vertices of V' are in the same connected component as our start vertex s
- ▶ This can only be improved by a constant factor

Application example

Image processing

- ▶ Connected component labeling
- ▶ Counting of objects in an image



Application example

Image processing

What's object, what's background?



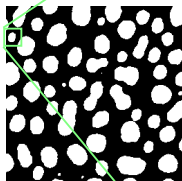
| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U |
|----|-----|----|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|----|----|----|-----|-----|-----|
| 35 | 104 | 80 | 56 | 40 | 16 | 16 | 8 | 16 | 16 | 24 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 24 | 24 | 16 | |
| 36 | 80 | 64 | 48 | 32 | 16 | 16 | 16 | 24 | 32 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 32 | 32 | 24 | 24 | 24 |
| 37 | 56 | 48 | 32 | 24 | 8 | 16 | 16 | 32 | 40 | 48 | 48 | 48 | 40 | 40 | 40 | 40 | 32 | 32 | 24 | 24 | 24 |
| 38 | 40 | 32 | 24 | 24 | 16 | 32 | 48 | 64 | 72 | 80 | 80 | 72 | 56 | 56 | 48 | 48 | 40 | 40 | 32 | 32 | 32 |
| 39 | 16 | 16 | 16 | 24 | 24 | 48 | 72 | 88 | 104 | 112 | 112 | 96 | 72 | 64 | 56 | 48 | 40 | 40 | 40 | 40 | 40 |
| 40 | 16 | 16 | 24 | 40 | 56 | 88 | 120 | 128 | 136 | 144 | 144 | 120 | 96 | 88 | 72 | 56 | 48 | 48 | 40 | 40 | 40 |
| 41 | 8 | 16 | 24 | 56 | 80 | 120 | 160 | 168 | 168 | 168 | 168 | 144 | 120 | 104 | 80 | 64 | 48 | 48 | 40 | 40 | 32 |
| 42 | 16 | 32 | 40 | 80 | 112 | 144 | 176 | 176 | 176 | 168 | 152 | 128 | 112 | 88 | 64 | 48 | 40 | 32 | 32 | 24 | |
| 43 | 24 | 40 | 56 | 96 | 136 | 160 | 184 | 184 | 176 | 176 | 168 | 152 | 136 | 112 | 88 | 64 | 40 | 32 | 24 | 24 | 16 |
| 44 | 40 | 56 | 80 | 112 | 152 | 168 | 184 | 184 | 176 | 176 | 168 | 152 | 136 | 112 | 80 | 64 | 40 | 32 | 16 | 16 | 16 |
| 45 | 48 | 72 | 96 | 128 | 160 | 176 | 184 | 184 | 176 | 176 | 168 | 152 | 136 | 104 | 72 | 56 | 32 | 24 | 8 | 16 | 16 |
| 46 | 48 | 72 | 96 | 136 | 168 | 176 | 192 | 192 | 184 | 184 | 176 | 160 | 136 | 104 | 72 | 56 | 32 | 24 | 16 | 24 | 32 |
| 47 | 48 | 72 | 96 | 136 | 168 | 184 | 192 | 192 | 192 | 192 | 184 | 160 | 136 | 104 | 72 | 48 | 24 | 24 | 16 | 32 | 48 |
| 48 | 48 | 72 | 96 | 128 | 168 | 184 | 200 | 200 | 200 | 192 | 184 | 160 | 128 | 96 | 64 | 48 | 24 | 32 | 32 | 56 | 72 |
| 49 | 48 | 72 | 88 | 128 | 160 | 184 | 200 | 200 | 200 | 192 | 184 | 152 | 120 | 88 | 56 | 40 | 24 | 32 | 40 | 72 | 96 |
| 50 | 48 | 64 | 80 | 112 | 136 | 160 | 176 | 176 | 176 | 168 | 160 | 136 | 104 | 80 | 48 | 40 | 32 | 40 | 56 | 88 | 128 |
| 51 | 48 | 64 | 72 | 96 | 112 | 128 | 144 | 152 | 152 | 144 | 136 | 112 | 88 | 64 | 40 | 40 | 32 | 48 | 64 | 112 | 152 |
| 52 | 48 | 56 | 64 | 80 | 88 | 104 | 112 | 112 | 120 | 112 | 104 | 88 | 72 | 56 | 32 | 32 | 32 | 64 | 88 | 128 | 168 |
| 53 | 40 | 48 | 48 | 56 | 64 | 72 | 72 | 80 | 80 | 80 | 72 | 64 | 48 | 40 | 24 | 32 | 32 | 72 | 104 | 144 | 184 |
| 54 | 48 | 48 | 48 | 48 | 48 | 56 | 56 | 56 | 64 | 56 | 56 | 48 | 40 | 32 | 24 | 40 | 48 | 88 | 128 | 160 | 200 |

Application example

Image processing

Convert to black white using threshold:

value = 255 **if** value > 100 **else** 0



| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|----|-----|---|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|
| 35 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 |
| 44 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 |
| 45 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 |
| 46 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 |
| 47 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 |
| 48 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 |
| 49 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 |
| 51 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 52 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 56 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 57 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 58 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Application example

Image processing

Interpret image as graph:

- ▶ Each white pixel is a node
- ▶ Edges between adjacent pixels (normally 4 or 8 neighbors)
- ▶ Edges are not saved externally, algorithm works directly on array
- ▶ Breadth- / depth-first search find all connected components (particles)

Application example

Image processing

Find connected components:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|----|-----|---|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|-----|-----|
| 35 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 47 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 48 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 49 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 51 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 |
| 52 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 |

- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1

Application example

Image processing

Find connected components:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|----|-----|---|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 35 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 40 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 41 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 42 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 43 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 44 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 45 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 46 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 47 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 48 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 49 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 50 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 51 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 52 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 |

- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels

Application example

Image processing

Find connected components:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|----|-----|---|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|---|-----|-----|
| 35 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 47 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 48 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 49 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 51 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 |
| 52 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 |

- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels
- ▶ Label non-zero pixels as component 1

Application example

Image processing

Find connected components:



- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels
- ▶ Label non-zero pixels as component 1

Application example

Image processing

Find connected components:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|----|-----|---|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|-----|-----|
| 35 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 47 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 48 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 49 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 51 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 |
| 52 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 |

- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels
- ▶ Label non-zero pixels as component 1

Application example

Image processing

Find connected components:



- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels
- ▶ Label non-zero pixels as component 1

Application example

Image processing

Find connected components:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|----|-----|---|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|-----|-----|
| 35 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 47 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 48 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 49 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 51 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 |
| 52 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 |

- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels
- ▶ Label non-zero pixels as component 1

Application example

Image processing

Find connected components:



- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels
- ▶ Label non-zero pixels as component 1

Application example

Image processing

Find connected components:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|----|-----|---|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|-----|-----|
| 35 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 47 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 48 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 49 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 |
| 51 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 |
| 52 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 |

- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels
- ▶ Label non-zero pixels as component 1

Application example

Image processing

Find connected components:



- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels
- ▶ Label non-zero pixels as component 1

Application example

Image processing

Find connected components:

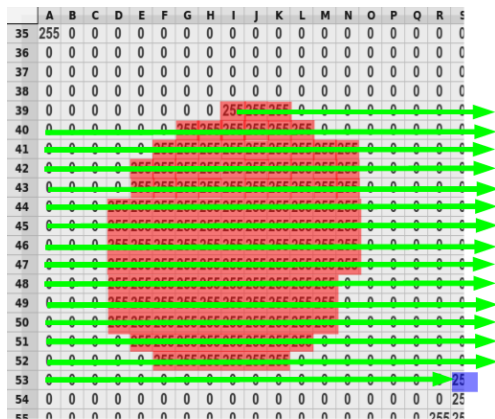
| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|----|-----|---|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|---|-----|-----|
| 35 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 47 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 48 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 49 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 |
| 51 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 |
| 52 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 |

- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 1
- ▶ Check neighbors of all new labeled pixels
- ▶ Label non-zero pixels as component 1

Application example

Image processing

Find connected components:



- ▶ Search pixel-by-pixel for non-zero intensity
- ▶ Label found pixel as component 2
- ▶ ...

Application example

Image processing

Result of connected component labeling:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X |
|----|-----|---|---|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|---|---|---|-----|-----|-----|-----|
| 35 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 25 |
| 47 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 25 |
| 48 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 25 |
| 49 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 25 |
| 50 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 |
| 51 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 |
| 52 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 56 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 57 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 58 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 59 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 60 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 61 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 62 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 63 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 64 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 65 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |
| 66 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 255 | 255 | 255 | 255 |



| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W |
|----|----|---|---|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|----|----|----|
| 35 | 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 47 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 |
| 48 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 |
| 49 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 |
| 50 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 |
| 51 | 0 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 |
| 52 | 0 | 0 | 0 | 0 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 54 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 55 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 56 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 57 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 58 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 59 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 60 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 61 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 62 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 63 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 64 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 65 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |
| 66 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 17 | 17 |

Figure: Result: particle indices instead of intensities

Further Literature

► General

[CRL01] Thomas H. Cormen, Ronald L. Rivest, and Charles E. Leiserson.

Introduction to Algorithms.

MIT Press, Cambridge, Mass, 2001.

[MS08] Kurt Mehlhorn and Peter Sanders.

Algorithms and data structures, 2008.

<https://people.mpi-inf.mpg.de/~mehlhorn/ftp/Mehlhorn-Sanders-Toolbox.pdf>.

Further Literature

► Graph-Search

[Wika] [Breadth-first search](https://en.wikipedia.org/wiki/Breadth-first_search)

[https://en.wikipedia.org/wiki/
Breadth-first_search](https://en.wikipedia.org/wiki/Breadth-first_search)

[Wikb] [Depth-first search](https://en.wikipedia.org/wiki/Depth-first_search)

[https:
//en.wikipedia.org/wiki/Depth-first_search](https://en.wikipedia.org/wiki/Depth-first_search)

► Graph-Connectivity

[Wik] [Connectivity \(graph theory\)](https://en.wikipedia.org/wiki/Connectivity_(graph_theory))

[https://en.wikipedia.org/wiki/Connectivity_
\(graph_theory\)](https://en.wikipedia.org/wiki/Connectivity_(graph_theory))