Karlsruhe Institute of Technology

Algorithms for graph visualization

Project - Orthogonal Grid Layout with Small Area

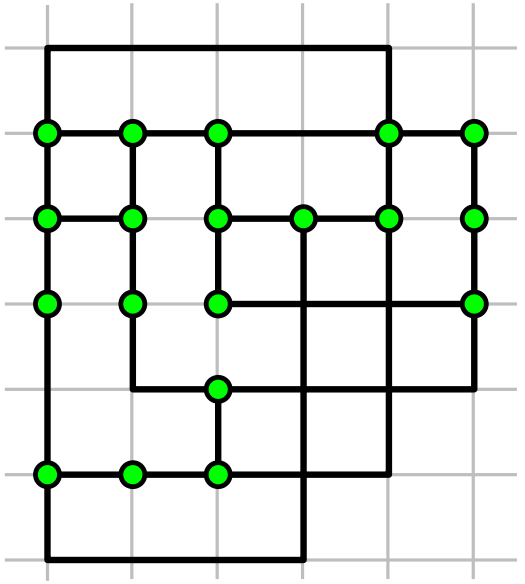
WINTER SEMESTER 2013/2014

Tamara Mchedlidze – Martin Nöllenburg

Orthogonal Grid Layout with Small Area

Let $G = (V, E)$ be a graph with maximum degree 4.
An **orthogonal grid layout** is a drawing of G such that:

- Vertices are on distinct grid points
- Edges are **orthogonal**, (sequence of vertical and horizontal segments)
- Bends lie on grid. No limit on the number of bends.
- Crossings are allowed. They occupy grid points. No limit on the number of crossings.
- Overlaps are forbidden.

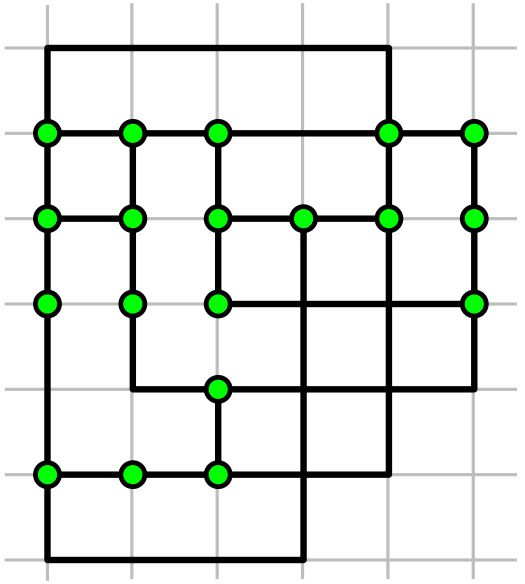


Area of orthogonal layout is the number of grid points in the smallest rectangle that encloses the layout.

Orthogonal Grid Layout with Small Area

Let $G = (V, E)$ be a graph with maximum degree 4.
An **orthogonal grid layout** is a drawing of G such that:

- Vertices are on distinct grid points
- Edges are **orthogonal**, (sequence of vertical and horizontal segments)
- Bends lie on grid. No limit on the number of bends.
- Crossings are allowed. They occupy grid points. No limit on the number of crossings.
- Overlaps are forbidden.



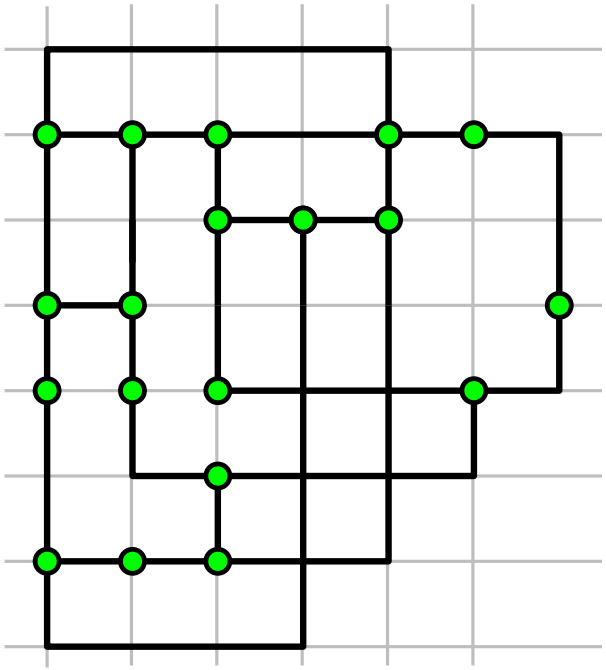
Area of orthogonal layout is the number of grid points in the smallest rectangle that encloses the layout.

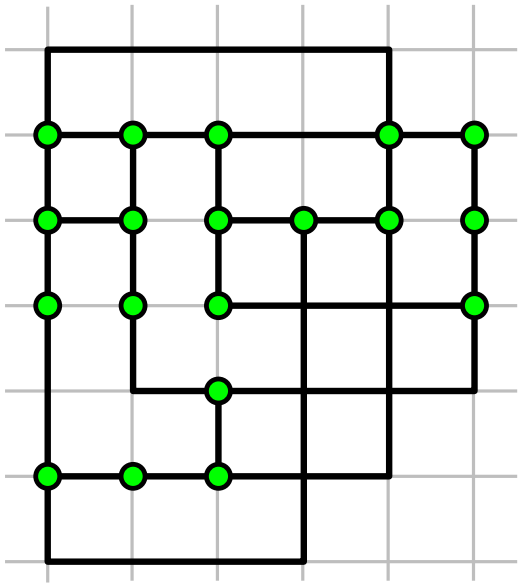
We are looking for orthogonal grid drawings with small area.

Orthogonal Grid Layout with Small Area

Given an orthogonal layout of G one can do the following to reduce the area:

- Move vertices
- Move edges
- Redraw/re-route edges





Orthogonal Grid Layout with Small Area

Karlsruhe Institute of Technology

Demo

Orthogonal Grid Layout with Small Area

- **Our target**: A program that takes as input a graph layout and outputs a layout with as small area as possible.

Orthogonal Grid Layout with Small Area

- **Our target**: A program that takes as input a graph layout and outputs a layout with as small area as possible.
- **Input and output** format:

```
5                                  ← Number of vertices

4 2                                ← Vertex 0
2 4
4 4
6 4
4 6

1 0 [ 2 2 ]                        ← Edge (1, 0)
0 3 [ 6 2 ]
2 1 []
0 2 []          Sequence of bend
1 4 [ 2 6 ]     coordinates
4 3 [ 6 6 ]          ↓
0 4 [ 4 0 0 0 0 8 4 8 ]
```

Orthogonal Grid Layout with Small Area

- **Our target**: A program that takes as input a graph layout and outputs a layout with as small area as possible.
- **Input and output** format:
- **What are you expected to do?**
  - Come up with your own algorithm, heuristics, interactive approach
  - Use existing algorithms, their combination, modifications

```
5                          ← Number of vertices

4 2                        ← Vertex 0
2 4
4 4
6 4
4 6

1 0 [ 2 2 ]                ← Edge (1, 0)
0 3 [ 6 2 ]
2 1 []
0 2 []          Sequence of bend
1 4 [ 2 6 ]     coordinates
4 3 [ 6 6 ]          ↓
0 4 [ 4 0 0 0 0 8 4 8 ]
```

Orthogonal Grid Layout with Small Area

- **Our target**: A program that takes as input a graph layout and outputs a layout with as small area as possible.
- **Input and output** format:
- **What are you expected to do?**
  - Come up with your own algorithm, heuristics, interactive approach
  - Use existing algorithms, their combination, modifications
- **What can you use?**
  - Any library freely available for academic use
  - Any graph drawing algorithm, whether included in the lecture material or not

```
5                                  ← Number of vertices

4 2                                ← Vertex 0
2 4
4 4
6 4
4 6

1 0 [ 2 2 ]                        ← Edge (1, 0)
0 3 [ 6 2 ]
2 1 []
0 2 []          Sequence of bend
1 4 [ 2 6 ]     coordinates
4 3 [ 6 6 ]          ↓
0 4 [ 4 0 0 0 0 8 4 8 ]
```

Some Available Tools

- **OGDF**- Open Graph Drawing Framework
  - C++ library with implementations of graph drawing algorithms
  - open-source, free
  - Contains an implementation of orthogonal layout algorithm
- **yFiles**
  - Java library with implementations of graph drawing algorithms
  - Version 2.9 can be provided
  - Implementation of orthogonal layout algorithm
  - Compaction algorithm for orthogonal layout
- **JUNG**-Java Universal Network/Graph Framework
  - Java library with algorithms for analysis and visualization of graphs
  - open-source, free
- **Gephi**-Interactive Visualization and Exploration Platform
  - open-source, free
  - Plugin using Gephi API
  - Existing plugin?

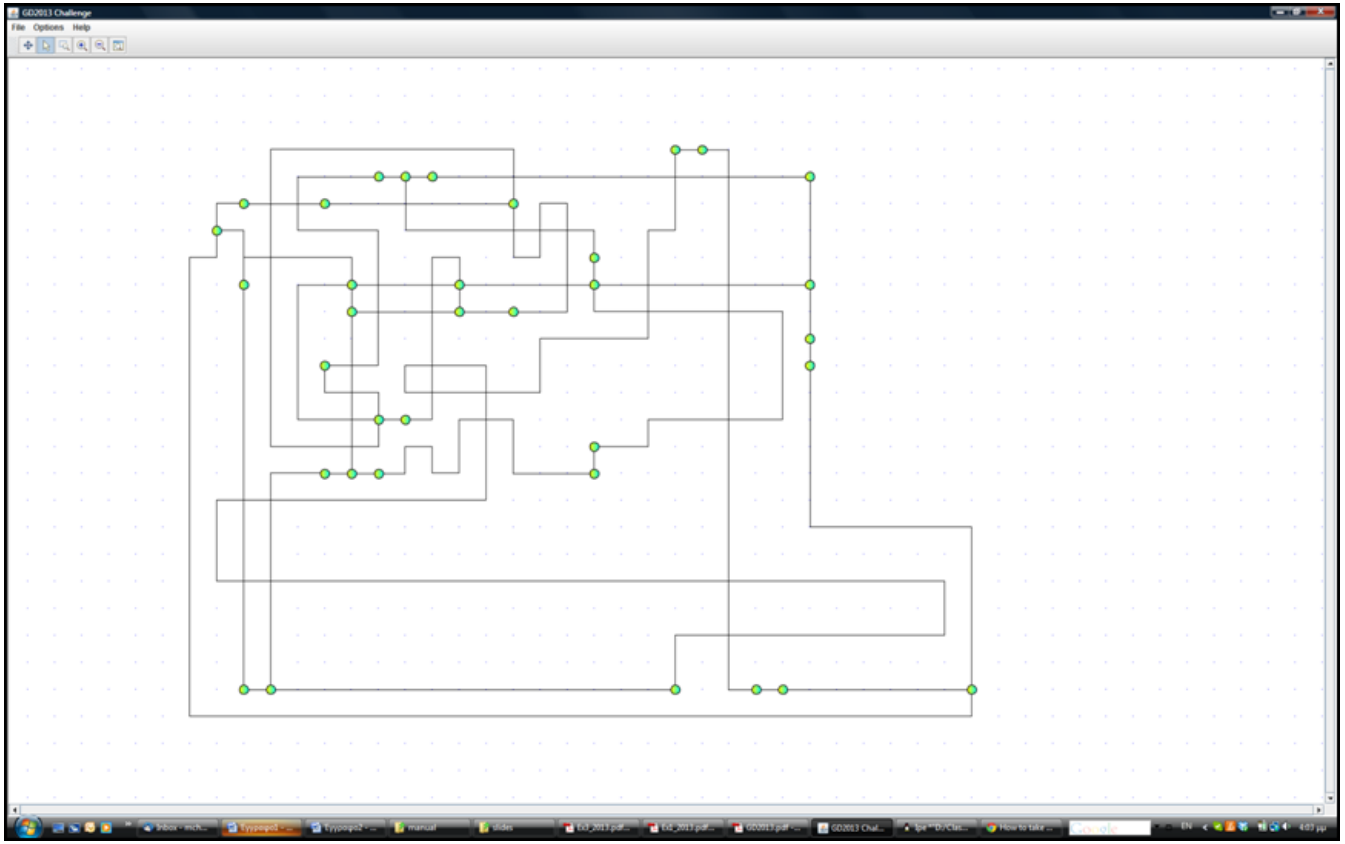
Data

- You will be provided with several data files to test your algorithms
- An application that displays this data format and allowes manual modifications

```
5

4 2
2 4
4 4
6 4
4 6

1 0 [ 2 2 ]
0 3 [ 6 2 ]
2 1 []
0 2 []
1 4 [ 2 6 ]
4 3 [ 6 6 ]
0 4 [ 4 0 0 0 0 8 4 8 ]
```



How to Start?

- Play with the provided tool and the graphs to create ideas on how to proceed

How to Start?

- Play with the provided tool and the graphs to create ideas on how to proceed
- Look for the existing implementations of orthogonal layout algorithms at the available libraries

How to Start?

- Play with the provided tool and the graphs to create ideas on how to proceed
- Look for the existing implementations of orthogonal layout algorithms at the available libraries
- Think which other algorithms can provide you an initial layout

Graph G ⟶ Initial Layout Γ ⟶ Orthogonalize Γ

How to Start?

- Play with the provided tool and the graphs to create ideas on how to proceed
- Look for the existing implementations of orthogonal layout algorithms at the available libraries
- Think which other algorithms can provide you an initial layout

Graph G ⟶ Initial Layout Γ ⟶ Orthogonalize Γ

- Think but have your mind open in the upcomming lectures to get further ideas!

&

Before You Start

- Form Groups of 2 to 4 people

Before You Start

- Form Groups of 2 to 4 people
- Schedule Meetings
  - **1st Meeting** - Discuss Ideas, Questions: **Before Christmas**
  - **2nd Meeting** - Finalize Ideas: **Mid January**
  - **3rd Meeting** - Intermediate Discussion: **Late January**
  - **4th Meeting** -1st Version: **February**
  - **5th Meeting** - Final Version, Presentation, Competition: **Late March**

Before You Start

- Form Groups of 2 to 4 people
- Schedule Meetings
  - **1st Meeting** - Discuss Ideas, Questions: **Before Christmas**
  - **2nd Meeting** - Finalize Ideas: **Mid January**
  - **3rd Meeting** - Intermediate Discussion: **Late January**
  - **4th Meeting** -1st Version: **February**
  - **5th Meeting** - Final Version, Presentation, Competition: **Late March**

Presentation

- 20 min presentation (including questions) of the ideas behind your approach

Before You Start

- Form Groups of 2 to 4 people
- Schedule Meetings
  - **1st Meeting** - Discuss Ideas, Questions: **Before Christmas**
  - **2nd Meeting** - Finalize Ideas: **Mid January**
  - **3rd Meeting** - Intermediate Discussion: **Late January**
  - **4th Meeting** -1st Version: **February**
  - **5th Meeting** - Final Version, Presentation, Competition: **Late March**

Presentation

- 20 min presentation (including questions) of the ideas behind your approach
- Competition:
  - Teams will receive a new collection of graphs
  - After one hour, all teams will submit their final drawings
  - The team with the highest cumulative score wins